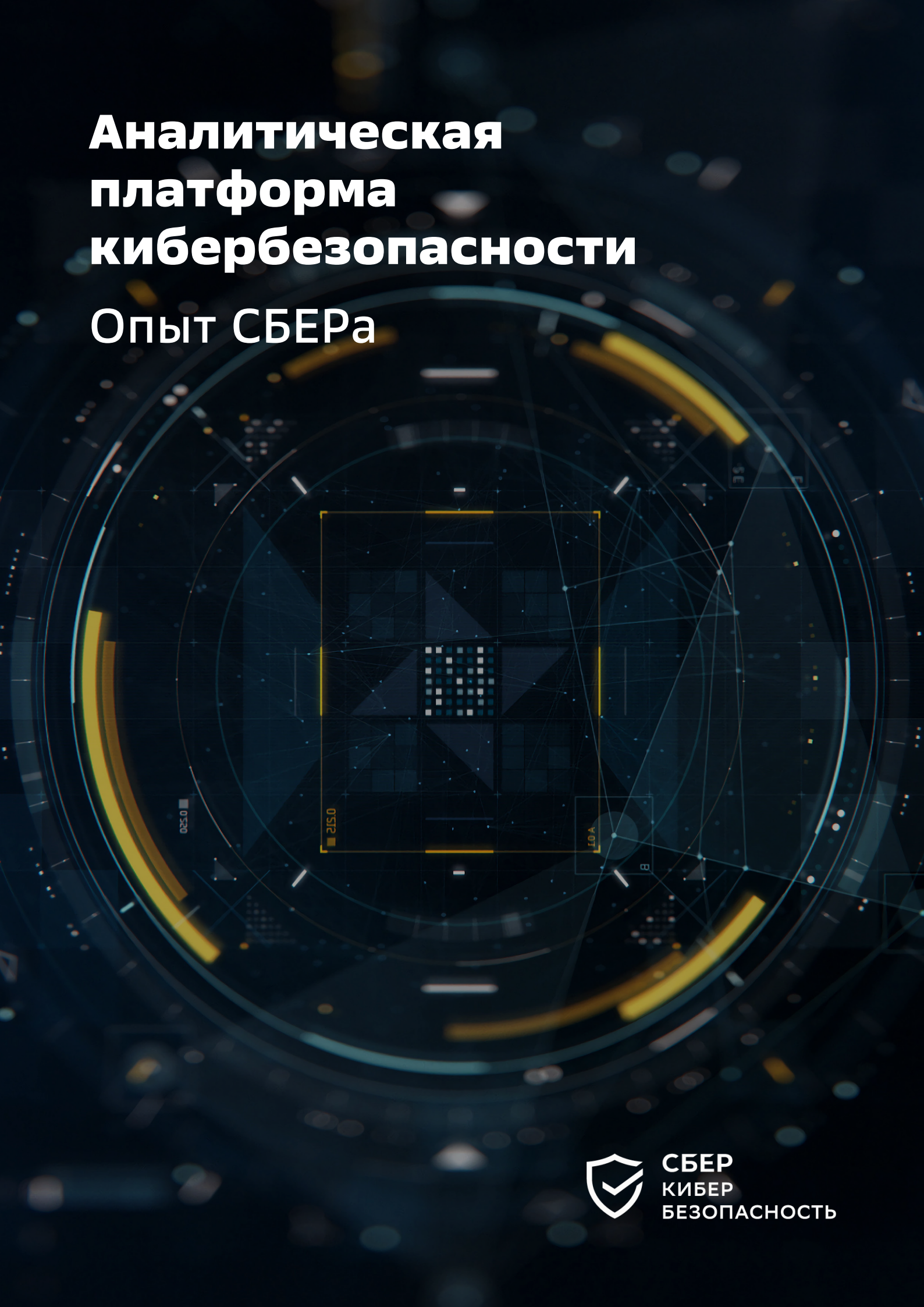
Аналитическая платформа кибербезопасности

Опыт СБЕРа



СБЕР
КИБЕР
БЕЗОПАСНОСТЬ

Уважаемые коллеги!

Достаточно давно финансовые организации стали целью номер один для киберпреступности, что связано как с развитием цифровых моделей ведения бизнеса, потенциалом быстрой монетизации для злоумышленников, так и с наличием большого числа клиентов.

Геополитическая ситуация, уход иностранных компаний, санкционное давление только усилили рост киберугроз. В этих условиях появилась необходимость объединять компоненты архитектуры кибербезопасности в единую систему для возможности оперативного управления рисками, опираясь на собственные решения.

Одной из ключевых компонент становятся платформы, обеспечивающие обработку событий в режиме реального времени и хранение всех данных в контексте киберезопасности, являющиеся фундаментом для создания и развития продуктов кибербезопасности.

Платформа кибербезопасности СБЕРа предназначена в первую очередь для решения задач по направлениям защиты инфраструктуры и антифрод. Платформа обрабатывает свыше 500 млрд событий в день от 535 типов источников, хранит 12 Пб данных, объем которых динамично увеличивается.

Надеемся, что представленные материалы помогут компаниям и экспертному сообществу в России для решения схожих задач.

Искренне ваш,

Кузнецов Станислав Константинович

Заместитель Председателя Правления ПАО Сбербанк

1. Введение

1.1 Предпосылки создания

За последние годы цифровые технологии существенно изменили структуру современных финансовых организаций, повысили их привлекательность, позволили использовать новые перспективные модели ведения бизнеса, которые в свою очередь спровоцировали новые всплески киберпреступности.

Геополитическая ситуация, уход иностранных компаний, санкционный режим и фактически начавшаяся киберагрессия привели к росту рисков кибербезопасности, связанных с прерыванием оказания банковских услуг, хищением средств клиентов, утечкой конфиденциальных данных. А также возникла критическая потребность оперативной замены целых классов инфраструктурных и прикладных решений от иностранных производителей.

Кибератаки стали сложнее, а их проведением начали заниматься не только отдельные группировки, но и целые сообщества обычных людей, объединенных идеологическими принципами. При этом координация атак может осуществляться спецслужбами различных государств, а производители могут активировать аппаратные «закладки» или предоставлять информацию об уязвимостях нулевого дня.

Появилась потребность в механизмах, объединяющих существующие средства защиты для комплексной работы с рисками кибербезопасности и принятии управленческих решений. Изначально развитие систем кибербезопасности в СБЕРе шло по пути закупки и развертывания коробочных решений, как правило, слабо связанных между собой и не обеспечивающих полной информации о контексте. Подобный фрагментарный подход обеспечивал надежную защиту от типовых угроз и обеспечивал возможность внедрения нового решения в кратчайшие сроки, но при этом не позволял увидеть полноценную картину происходящего в ИТ-ландшафте, оценить риски и обеспечить их минимизацию (Рисунок 1).

В 2019 году в СБЕРе создана платформа кибербезопасности, которая включает набор продуктов и технологических сервисов,

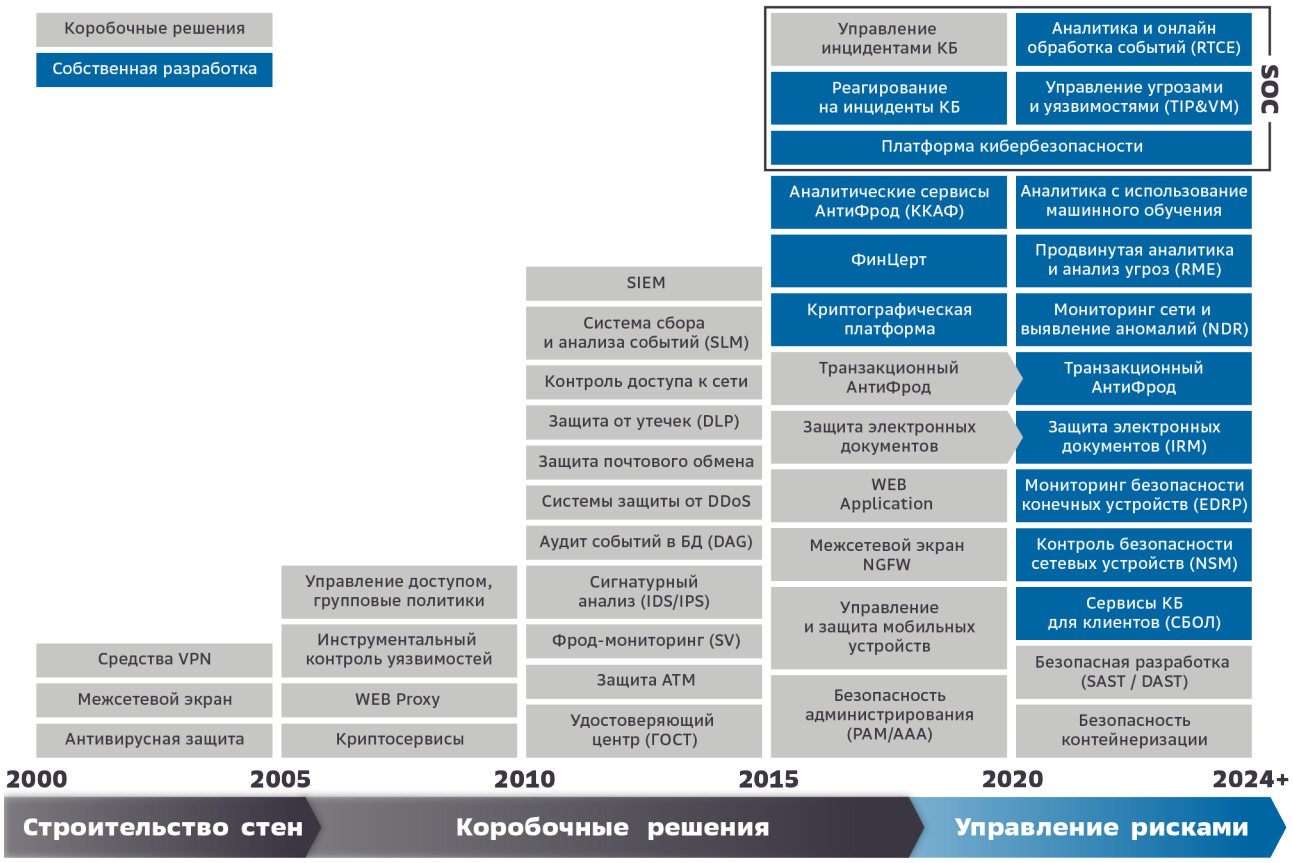


Рисунок 1. *Эволюция развития систем защиты Банка*

В 2023 ГОДУ ПЛАТФОРМА КИБЕРБЕЗОПАСНОСТИ СБЕРА: ОБРАБАТЫВАЕТ ПОТОК БОЛЕЕ 500 МЛРД. СОБЫТИЙ В СУТКИ, ОБЪЕМ ХРАНИЛИЩА СОСТАВЛЯЕТ 25 ПБ, ИСПОЛЬЗУЕТ СВЫШЕ 100 AI-МОДЕЛЕЙ

предназначенных для: противодействия внутреннему и внешнему мошенничеству, защиты инфраструктуры, мониторинга и анализа угроз. Платформа позволяет создавать аналитические продукты, работать с большими данными и AI-моделями. Технологические сервисы платформы позволяют объединять существующие средства защиты и обеспечивают: сбор, хранение и обработку событий от различных источников в режиме реального времени; быструю адаптацию моделей и алгоритмов к новым, постоянно изменяющимся угрозам; проведение расследований.

1.2 Цель документа

Целью создания данного документа является систематизация знаний и обмен опытом с экспертным сообществом в задачах построения аналитических платформ кибербезопасности.

Предлагаемый документ «Аналитическая платформа кибербезопасности, опыт СБЕРа» содержит концептуальную архитектуру, основные принципы построения платформенных решений, состав функциональных компонент и их назначение, а также сравнение различных платформенных решений между

собой, которое может служить основой для построения технических решений для задач кибербезопасности.

Материалы могут стать основой для создания открытого сообщества разработчиков систем кибербезопасности с применением технологий прикладной, продвинутой, прогнозной и предиктивной аналитики. Материалы не являются эталонной моделью и методическим руководством для создания систем такого класса, но могут быть использованы в качестве примера для построения аналогичных систем.

1.3 Тенденции в области кибербезопасности

Лавинообразное развитие технологий, цифровизация современного общества, дефицит кадров в сфере ИТ и эскалация международной напряженности приводит к существенному увеличению рисков в области кибербезопасности. Злоумышленники продолжают совершенствовать свое мастерство, разрабатывая новые техники атак с использованием возможностей искусственного интеллекта, LLM моделей, программ вымогателей и механизмов социальной инженерии.

Идет активное применение бизнес-модели «вредоносное ПО как услуга» (Malware-as-a-Service, MaaS), когда злоумышленники за определенную плату предоставляют доступ к вредоносному программному обеспечению и соответствующей инфраструктуре управления. Анализ Bi.Zone[1] показывает, что использование вредоносного программного обеспечения по модели Malware-as-a-Service позволяет злоумышленникам, которым не хватает компетенций для собственной разработки, атаковать даже крупные организации. Среди основных трендов нового десятилетия можно выделить следующие:

Цифровизация общества и 3-кратный рост рынка электронной коммерции повлечет за собой увеличение объемов мошенничества, рост успешных фишинговых атак и атак на биометрию, угроз репутации публичных лиц. По мнению аналитиков СБЕРа к 2026 г. объем реализованного мошенничества в России может составить 20 млрд руб., к 2030 г. – 27 млрд руб.

Эскалация международной напряженности влечет за собой рост киберинцидентов и изменение формата киберугроз. В новых условиях целью атак являются приостановка деятельности крупных организаций и удары по их репутации,

[1] ***Bi.Zone:*** *семь ликов тьмы*

а не обогащение злоумышленников.

Запрет на доступ к зарубежным базам уязвимостей, данным об индикаторах компрометации и официальных обновлений влечет за собой «ослабление» средств защиты, что в свою очередь увеличивает вероятность реализации успешных кибератак на небольшие (дочерние) компании.

Количество утечек персональных данных продолжает расти, при этом злоумышленники изменили тактику: вместо продажи данных в даркнет или шантажа пострадавшей организации перешли к массовой публикации этих данных в открытом виде в мессенджерах и на форумах для нанесения репутационного и экономического ущерба бизнесу.

Анализ Bi.Zone[2] показывает, что в 2023 году в 20% киберинцидентов злоумышленники получали первоначальный доступ к инфраструктуре с помощью легитимных учетных данных. За первые 10 месяцев 2023 информация об отечественных компаниях попадала на теневые форумы в три раза чаще, чем за аналогичный период 2022 года.

Искусственный интеллект становится мощным оружием в руках злоумышленников, так как он может быть использован для поиска новых уязвимостей, генерации эксплойтов, созданию новых схем социальной инженерии и дипфейков, созднию вирусного программного обеспечения.

Рост числа кибератак и постоянно меняющийся формат угроз является одним из ключевых вызовов нашего времени. Сложность кибератак постоянно растет, увеличивается количество новых уязвимостей и расширение «окна уязвимостей». Одновременно с этим происходит существенное «ослабление» средств защиты вследствие потери знаний о новых киберугрозах.

Аналитики отмечают, что если в начале 2023 года основными целями хакеров были финтех-сфера, ритейл и игровая индустрия, то во втором квартале 2023 они переключили свое внимание на отрасли, связанные с здравоохранением, образованием, логистикой и даже сельским хозяйством. При этом вредоносная активность нацелена в первую очередь на оказание деструктивного воздействия, а не получение выгоды.

Прогнозы аналитических компаний всего мира указывают, что крупные организации не смогут минимизировать угрозы исключительно за счет оптимизации существующих процессов

[2] ***Bi.Zone:*** *каждая пятая атака происходит с использованием утекших данных пользователей*

В 2023 ГОДУ СБЕР УСПЕШНО ОТРАЗИЛ

124 **DDOS-АТАК**

1 **МЛН. ЗАПРОСОВ/СЕК. СОСТАВИЛА САМАЯ МОЩНАЯ АТАКА**

30%

Предприятий перейдут на платформы, обеспечивающие высокий уровень безопасности и возможности быстрого расширения функциональности к 2025 году *(Gartner)*

и незначительной модернизации используемых технологий. Текущие разрозненные системы защиты перестают справляться с новыми типами атак. Возникает необходимость централизованного анализа данных от всех источников и непрерывной оценки угроз.

Платформы кибербезопасности становятся одним из ключевых направлений развития крупных компаний. Они обеспечивают консолидацию и интеграцию всех технических решений в единое рабочее пространство, обмен событиями в режиме реального времени, поддержку прогнозной и предиктивной аналитики.

1.4 Переход от коробочных решений к платформам управления рисками

Использование коробочных решений для конкретных задач кибербезопасности обладает определенными преимуществами. Например, наличие на рынке компаний-интеграторов, обладающих собственным штатом специалистов с высоким уровнем компетенций, позволяет оперативно закрыть конкретную потребность.

К основным минусам коробочных решений можно отнести сложность интеграции решений разных производителей между собой. Обмен данными между разными решениями может быть затруднен из-за разного формата данных и отсутствия универсальных интеграционных коннекторов, что в свою очередь несет трудности при комплексном анализе событий.

Эволюция систем сбора и корреляции событий безопасности шла по пути постепенного объединения различных технологий в рамках централизованного решения:

1. Система управления журналами событий (Log Management, LM).
2. Система управления событиями безопасности (Security Log Management, SLM).
3. Система сбора, анализа и корреляции событий безопасности (Security information and event management, SIEM).
4. Ситуационный центр кибербезопасности (Security Operations Center, SOC).
5. Система управления рисками кибербезопасности.

По мере того, как инструменты кибербезопасности становят-

ся более интеллектуальными, они все больше зависят от качества, полноты и достоверности данных. Возникает потребность в обмене данными между отдельными техническими решениями, создании аналитических систем, которые обеспечивают агрегацию и обработку всех событий в режиме реального времени, быструю адаптацию моделей и алгоритмов к новым, постоянно изменяющимся угрозам, анализ рисков.

Одним из самых актуальных направлений развития инструментов кибербезопасности становятся платформенные решения, которые позволяют объединять в единое целое мониторинг, анализ и контроль всех источников данных. Благодаря интеграции всех компонент в едином унифицированном ландшафте, платформы кибербезопасности получают возможность анализировать и коррелировать данные, полученные из различных разрозненных источников, формировать единое видение текущей обстановки, быстро выявлять угрозы и реагировать на них.

Платформа — это логическое объединение программных продуктов, а также методология, документация, фреймворки, предназначенные для создания аналитических инструментов кибербезопасности. Объединение и обработка информации от различных коробочных решений в рамках платформенных сервисов позволяет решать сложные аналитические задачи, выявлять новые риски, которые невозможно определить при использовании отдельных инструментов.

Платформы могут служить основой как для создания автоматизированных систем, так и для объединения их в рамках единой функциональной области. Автоматизированная система — это бизнес-приложение, представляющее собой набор разработанных или заимствованных компонентов, реализующих логически связанный набор функциональности, для выполнения локальных задач.

Платформы для хранения, обработки и аналитики данных являются связующим элементом для всех инструментов кибербезопасности компании

Платформенные решения в области кибербезопасности обладают несколькими явными преимуществами: снижение совокупной стоимости за счет переиспользования компонентов; уменьшение времени внедрения, устранение зависимости от конкретного вендора; возможность быстрого создания новых продуктов кибербезопасности на основе набора уже готовых модулей; а также построение экосистем кибербезопасности на основе данных платформы.

2. Платформа кибербезопасности

2.1 Цели и задачи аналитической платформы кибербезопасности

Ключевая задача Платформы Кибербезопасности – объединить все существующие элементы защиты организации в единое целое, стандартизировать архитектуру и набор технологий, автоматизировать рутинные процессы, упростить создание новых продуктов.

Целью создания аналитической платформы кибербезопасности является объединение всей информации, поступающей от средств защиты, за счет централизованной загрузки, хранения, обработки, анализа, выявления корреляционных зависимостей, что обеспечит повышение осведомленности об угрозах, предотвращение нарушений политик, проактивное реагирование, прогнозирование потенциальных потерь.
Первоочередной задачей, на решение которой направлено создание аналитической платформы, является обеспечение сервисов кибербезопасности необходимыми данными. При этом, основными источниками данных являются как инструменты кибербезопасности, так и инфраструктурные компоненты, которые предоставляют данные в разнообразных форматах. Для обеспечения простоты интеграции с источниками данных необходимо стандартизировать архитектуру и набор технологий, упростить процесс создания механизмов интеграции и приложений, а также автоматизировать рутинные процессы.
Платформа кибербезопасности, разработанная СБЕРом, объединяет в себе набор продуктов и сервисов, предназначенных для противодействия внутреннему и внешнему мошенничеству, защиты инфраструктуры, мониторинга и анализа угроз.

2.2 Функциональность платформы кибербезопасности

Функциональность аналитической платформы кибербезопасности должна включать сервисы для агрегации, нормализации и корреляции событий в режиме реального времени, инструменты для проведения аналитики, интеграционные компоненты для взаимодействия технических решений между собой; DevOps конвейер для быстрого создания, развер-

тывания новых компонентов.

Аналитическая платформа состоит из продуктов кибербезопасности и технологических сервисов.

Продукты кибербезопасности — решения собственной разработки, которые создаются с использованием технологических сервисов и предназначены для обеспечения защиты информационных активов организации и экосистемы. Например: антифрод, система корреляции событий, мониторинг сети и выявление аномалий, безопасность конечных устройств, управление угрозами и уязвимостями.

Технологический сервис — компонент платформы, который реализует набор функций, с целью их дальнейшего переиспользования продуктами кибербезопасности, снижая риски и затраты реализации. Технологический сервис — это атомарная единица, используемая в роли «строительных блоков» при построении приложений. Использование технологических сервисов позволяет командам разработки обеспечить требования архитектуры, надежности, безопасности и сопровождения для своих прикладных продуктов.

ТЕХНОЛОГИЧЕСКИЕ СЕРВИСЫ ПЛАТФОРМЫ МОЖНО РАССМАТРИВАТЬ КАК НАБОР ТИПОВЫХ СТРОИТЕЛЬНЫХ БЛОКОВ ДЛЯ СОЗДАНИЯ НОВЫХ АНАЛИТИЧЕСКИХ ПРОДУКТОВ ЛЮБОЙ СЛОЖНОСТИ

Технологические сервисы обеспечивают: сбор, агрегацию, нормализацию, парсинг событий в реальном времени и в пакетном режиме; интеграцию компонентов и взаимодействие с внешними системами; хранение и обработку данных; управление поставкой данных; разграничение доступа; аналитику, машинное обучение и возможность исполнения AI моделей.

Технологические сервисы составляют ядро платформы и разделены на четыре функциональные области в соответствии с реализуемой задачей (Рисунок 2):

- интеграционный слой;
- хранение и обработка;
- управление и контроль;
- аналитика и машинное обучение.

Интеграционный слой обеспечивает взаимодействие между различными компонентами, интеграцию с источниками данных и решениями кибербезопасности. Сервисы данной функциональной области включают типовые интеграционные адаптеры, позволяющие сторонним системам взаимодействовать с платформой, несмотря на различия в их структурах, форматах данных и протоколах. Использование

Рисунок 2. *Функциональные области аналитической платформы кибербезопасности*

интеграционного слоя позволяет решить проблемы несовместимости приложений и масштабируемости платформы, а также позволяют изменять отдельные элементы платформы без влияния на другие компоненты.

Хранение и обработка обеспечивает централизованное хранение структурированных и неструктурированных данных, загружаемых как в потоковом, так и в пакетном режиме, обработку данных с применением модели распределенных вычислений, предоставление вычислительных ресурсов для построения витрин, предоставление всех необходимых инструментов для работы с данными.

Управление и контроль включает механизмы управления доступом, контроль качества данных, управление метаданными, кроссплатформенный мониторинг, сервисы телеметрии, инструменты обеспечения безопасности платформы. В случае облачного развертывания аналитической платформы в данную функциональную область должны входить инструменты для управления ИТ-ландшафтом, программным и аппаратным обеспечением.

Аналитика и машинное обучение включает набор решений и технологий, обеспечивающих централизованный доступ к единому хранилищу данных, поиск информации по всем источникам хранилища, проведение исследований и разработку моделей, управление поставкой данных и исполнение

моделей, визуализацию отчетности и формирование дашбордов.

Для быстрого вывода продуктов и технологических сервисов в промышленную эксплуатацию требуется полноценный DevOps конвейер, обеспечивающий автоматизацию всех этапов жизненного цикла разработки.

Помимо типовых инструментов DevOps, обеспечивающих управление требованиями и релизами, версионное хранение кода, инфраструктуру сборки и развертывания приложений (CI\CD), сервисы разработчика должны включать в себя возможности проверки гипотез на ранней стадии. Особое внимание требуется уделить контролю уязвимостей и анализу безопасности исходного кода (SAST/DAST).

Практика СБЕРа показывает, что, используя комбинацию технологических сервисов, можно закрыть большинство потребностей аналитика информационной безопасности и построить продукт практически любой сложности за короткое время. Унификация механизмов интеграции и наборы готовых адаптеров позволят подключить практически любой источник, требующийся для решения задач кибербезопасности.

2.3 Обзор рынка

На рынке представлено достаточно большое количество производителей платформенных сервисов и решений, такие производители как IBM, Cisco, Check Point, Trend Micro, Lacework, Kaspersky, BI.ZONE, Positive Technologies, R-Vision предлагают набор своих продуктов и сервисов как единую платформу, обеспечивающую расширенные возможности для сбора данных из различных источников и их централизованный анализ.

В большинстве своем, производители предлагают готовые решения с множеством функций. Каждое из решений отлично закрывает свою определенную нишу, но при этом создает миллионы неструктурированных разнородных событий, которые достаточно трудно обрабатывать и анализировать. При этом, каждый производитель решения использует собственный формат данных и классификацию событий.

Разработчики технологических решений все чаще делятся на «портфельные» и «платформенные» направления:

- первые упаковывают «коробочные» продукты, объединяя их вокруг одного (или нескольких) ключевых решений, добавляют к ним инструменты централизованного управления и механизмы интеграции со сторонними решениями на основе открытых API;
- вторые объединяют инструменты в едином унифицированном ландшафте, обеспечивают их работу с единым аналитическим хранилищем (data lake), реализовывают набор технологических сервисов, обеспечивающих простоту интеграции и возможности по созданию новых продуктов и сервисов.

В настоящее время платформенные решения входят в линейку продуктов практически всех крупных поставщиков программного обеспечения. Их основу могут составлять решения класса SIEM, продукты для защиты конечных точек, сетевой инфраструктуры, сканирования инфраструктуры на уязвимости и SOAR решения.

Одним из наиболее популярных архитектурных подходов в области кибербезопасности является Gartner — Cybersecurity Mesh

Architecture (CSMA)[3]. В рамках данного подхода предлагается объединить все сервисы кибербезопасности в единую информационную среду для обмена информацией между различными средствами защиты.

Подход Cybersecurity Mesh Architecture (CSMA) включает следующие слои:

- агрегация и нормализация событий;
- интеллектуальный уровень обработки событий;
- идентификация;
- управления политиками и сценариями реагирования;
- консолидированные аналитические дашборды.

Связующим звеном для всех слоев подхода CSMA может являться аналитическая платформа кибербезопасности, которая обеспечивает сбор, агрегацию и нормализацию событий и предоставляет инструменты для интеллектуальной обработки. Функциональность платформы кибербезопасности может закрывать первые два слоя подхода CSMA.

Основной проблемой, решаемой CSMA, является окончательное размытие периметра организации, связанного с распространением удаленного формата работы и активным использованием облачных сервисов. Вместо выстраивания единого периметра, в пределах которого сосредоточены все защищаемые активы, CSMA предлагает формирование отдельных сегментов безопасности с едиными требованиями и политиками безопасности вокруг каждого актива, вне зависимости от его физического местоположения: корпоративная сеть, публичная сеть или внешнее облако.

По данным аналитического агентства KBV Research[4], основными игроками рынка, развивающими решения в соответствии с подходом CSMA, являются Fortinet, Checkpoint и IBM. Помимо них на данном рынке отмечены такие производители, как PaloAlto Networks, ForcePoint и Zscaler. При этом, ни один из производителей пока не может предоставить целостного предложения, закрывающего все слои концепции CSMA (Рисунок 3).

Fortinet позиционирует свои решения FortiSIEM и FortiSOAR как ключевой элемент платформы Fortinet Security Fabric. Они обеспечивают функционирование слоя аналитики безопасности и слоя информационных панелей подхода CSMA,

[3] ***Gartner: The Future of Security Architecture:*** *Cybersecurity Mesh Architecture (CSMA)*

на 90% снизят финансовые последствия инцидентов безопасности к 2024 году организации, использующие CSMA

[4] ***KBV Cardinal Matrix –*** *Cybersecurity Mesh Market Competition Analysis*

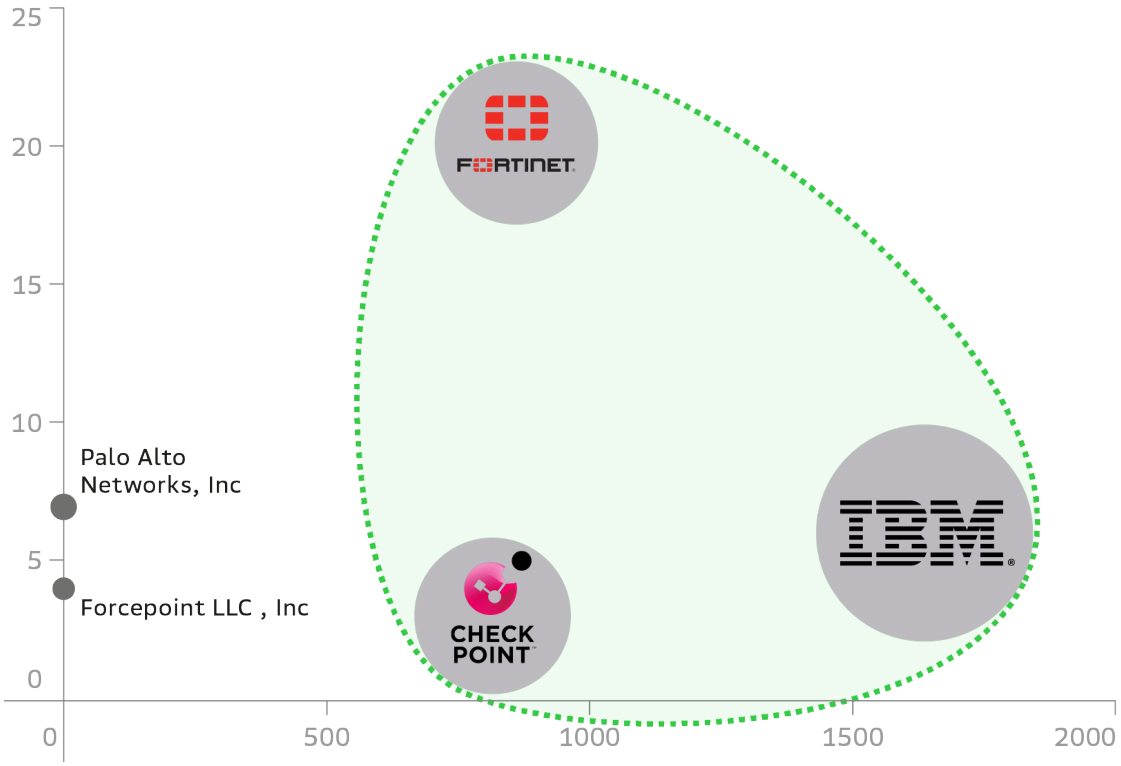


Рисунок 3. *Компании, развивающие продукты в соответствии с подходом CSMA*

в то время как закрытие других потребностей в области кибербезопасности может осуществляться за счет гибкой интеграции как с другими решениями Fortinet, так и с обширной экосистемой решений более 100 других производителей средств кибербезопасности и поставщиков фидов Threat Intelligence.

Checkpoint в свою очередь предлагает выстраивание платформы кибербезопасности в соответствии с принципами CSMA на базе продуктов класса XDR (eXtended Detection and Response) или MDR (Managed Detection and Response) объединённых общим продуктовым названием CheckPoint Horizon. MDR предоставляется в виде готового сервиса, где мониторинг и реагирование будет выполняться специалистами Checkpoint, в то время как XDR является on-premise решением.

Также в Checkpoint Horizon входит TI-сервис ThreatCloud, и вместе с XDR или MDR они обеспечивают функционирование слоя аналитики безопасности архитектуры CSMA. Слои консолидированных политик и информационных панелей обеспечиваются решением CheckPoint Infinity, отвечающим за централизованное управление всеми решениями Checkpoint, размещенными

в инфраструктуре, а также за отчетность и визуализацию событий и инцидентов кибербезопасности. В отличие от Fortinet Checkpoint делает больший упор на использование собственных продуктов, но также предоставляет возможности для интеграции решений других производителей.

IBM формирует свой подход к безопасности на основе четырех групп решений: инструменты обнаружения и реагирования на угрозы, агрегированные вокруг SIEM IBM Qradar; сервисы по защите мобильных устройств и конечных точек на основе продукта IBM Maas360; сервисов по поставке TI-фидов IBM X-Force Threat Intelligence; механизмы по управлению доступом Identity and Access Management (IAM) и Privileged Access Management (PAM).

В отличие от других производителей, рассмотренных выше, IBM предлагает собственные решения для слоя идентификации в архитектуре от CSMA. Также IBM делает большой акцент на защиту данных, которая не выделяется в отдельный слой в архитектуре CSMA.

Следует отметить, что подход CSMA призван решить проблему высоких издержек, связанных с эксплуатацией большого количества разрозненных решений кибербезопасности (число которых в крупных компаниях может доходить до 100), а также повысить качество выявления инцидентов и скорость реакции на них. При этом аналитики Gartner отмечают, что подход CSMA находится на ранней степени зрелости, а активное его развитие ожидается в горизонте 10 лет.

3. Архитектура платформы КБ

3.1 Инженерные задачи

Для построения корректной архитектуры платформы были собраны требования и выделены основные инженерные задачи, требующие детальной проработки с архитектурной и технической точки зрения.

При анализе задач, стоящих перед кибербезопасностью СБЕРа, было выявлено, что планируемый поток будет составлять не менее 6 млн событий в секунду (120 Тб данных в день), а прогнозируемый объем хранилища составит более 25 Пб с возможностью дальнейшего масштабирования.

Для обработки и хранения такого потока событий готовых решений не существует. При проектировании платформы кибербезопасности командой СБЕРа было выделено девять основных инженерных задач, для решения которых необходимо провести исследования и тестирования, проработать комбинацию из технических решений, позволяющих свести к минимуму собственную разработку.

Организовать прием и транспортировку 6 млн событий в секунду в режиме реального времени. Для решения данной задачи были определены и развернуты централизованные точки приема событий, определены и стандартизированы компоненты сбора и передачи, выявлены базовые конфигурации сервисов для обеспечения необходимой производительности.

Разработать модель данных для единообразной обработки, хранения и поиска по событиям. Сырые данные представляют собой в большинстве случаев неструктурированные логи, возникает необходимость приведения их к единой структуре. Была разработана базовая модель нормализации данных, которая позволяет определить ключевые поля, определить агрегирующие функции и временное окно.

Разработать универсальный механизм парсинга данных различных форматов и из любых типов источников. Раз-

Платформа кибербезопасности – одна из ключевых технологических составляющих SOC.

Какие бы ни были в SOC отлаженные процессы, но без удобного инструмента по работе с данными эффективность реагирования и аналитики будет близка к нулю. Благодаря платформе, SOC может предотвращать самые сложные инциденты.

Мнение заказчика

работан универсальный парсер, т.е. приложение, которое умеет обрабатывать различные форматы неструктурированных входных данных, превращая их в конечную структуру в виде JSON. Правила нормализации любого источника для универсального парсера представлены метаданными. На текущий момент универсальный парсер используется для обработки 535 типов источников.

Подготовить данные для автоматизированной аналитики кибербезопасности. Создан комплекс инструментов, обеспечивающий формирование онлайн и оффлайн фичей для машинного обучения и оркестрацию исполнения on-line и off-line моделей. Feature Store позволяет формировать витрины, являющиеся результатом агрегации исходных данных. На текущий момент Feature Store обрабатывает 10 тысяч аналитических витрин с ежедневным расчётом агрегатов, которые используют более чем 100 моделей.

Описать все используемые данные и обеспечить сквозное взаимодействие всех компонент платформы по метаданным. Реализован функционал создания реестра событий кибербезопасности с их описанием и категоризацией. Используя информацию о физической структуре данных источников, формируются уникальные настройки правил выявления событий различных типов в разрезе источников. Обеспечена сквозная интеграция всех компонентов платформы по метаданным. На текущий момент описано 27 тысяч таблиц и 700 тысяч полей.

Обеспечить одновременный аналитический поиск по различным типам хранилищ за приемлемое время. Реализован гетерогенный механизм доступа к данным с использованием универсального SQL синтаксиса, созданы дополнительные структуры индексирования в хранилище потоковых данных, обобщены и абстрагированы наиболее часто используемые подходы к выборке данных.

Организовать мониторинг всех компонентов платформы, состояния источников и данных. Источники: около 2 млн конечных точек. Создан многоуровневый мониторинг, включающий: инфраструктурный контроль, анализ качества данных, выявление аномалий в поведении источников, синтетический мониторинг скорости загрузки и обработки со-

бытий. Итогом работы многоуровневого мониторинга является профилирование источников с определением потерь и выявлением отклонений от ретроспективы.

Организовать защиту всех компонентов платформы и управления доступом к данным. Основная сложность обеспечении защиты платформы состояла в разрозненном технологическом стеке, включающем более 20 OpenSource компонентов, развернутых на 6000 нодах. Масштабы платформы, построенной из OpenSource компонентов, требуют применения нетривиальных методов защиты, что привело к необходимости разработки гибридных решений, состоящих из общедоступных OpenSource компонентов и элементов защиты собственной разработки.

Реализовать удобный интерфейс для задач кибербезопасности и Low code в технологических сервисах. В ходе анализа пользовательского опыта был спроектирован UI-кит, которой в дальнейшем лег в основу пользовательских интерфейсов всех продуктов и технологических сервисов. Для снижения нагрузки на разработчиков был принят подход к использованию Low code.

Детальное описание решений представленных выше технологических задач представлено в разделе 4.

3.2 Концептуальная архитектура аналитической платформы

Архитектура аналитической платформы должна решить проблему растущей сложности управления существующими системами кибербезопасности, обеспечить взаимодействие отдельных решений за счет стандартизации механизмов интеграции и оркестрации их работы с учетом текущих технологических ограничений.

Архитектура платформы спроектирована для возможности обработки большого потока событий безопасности и платежных операций в режиме реального времени.

Архитектура платформы кибербезопасности позволяет специализированным продуктам акцентировать свою разработку исключительно на профильной функциональности. Решение типовых задач полностью обеспечивается технологическими сервисами платформы, которые вызываются

Рисунок 4. *Высокоуровневая архитектура аналитической платформы КБ*

по мере функциональной необходимости (Рисунок 4).

Технологические сервисы реализуют универсальные, системные функции, не содержащие специфичной для прикладных продуктов логики. Они решают задачи загрузки, парсинга, доступа к данным, интеграции, управления, формирования отчетов и дашбордов.

Ключевым свойством технологических сервисов является возможность их переиспользования. Без повторного использования технологический сервис был бы простым архитектурным решением, которое было сделано один раз для решения определенной задачи.

Аналитическая платформа кибербезопасности состоит из нескольких основных технологических сервисов, обеспечивающих работу с данными.

Потоковая загрузка представляет собой набор инструментов для оперативного подключения к аналитической платформе потоковых источников данных и предназначен для

их online обработки, нормализации, парсинга. Сервис должен обеспечивать валидацию и нормализацию загружаемых на потоке данных и предоставлять данные в потоковом режиме для online-анализа.

Пакетная загрузка предназначена для получения данных по расписанию, при этом данные загружаются batch-ами с задержкой по времени от момента их появления в системе источнике и используются для offline обработки. Процесс загрузки включает в себя трансформацию данных, парсинг, обогащение.

Интеграционный сервис представляет набор инструментов для организации информационного взаимодействия между компонентами платформы и включает в себя распределенную среду потоковой передачи событий для обмена сообщениями на базе Apache Kafka и приложения в спецификации OpenAPI.

Оперативное хранилище предназначено для хранения данных, которые используются в поисковых и агрегационных запросах в режиме реального времени.

Аналитическое хранилище обеспечивает длительное хранение структурированных и неструктурированных данных для выполнения различных аналитических расчётов, обучения моделей, а также в качестве архива.

Сервис доступа к данным предоставляет единый программный API для сервисов и продуктов платформы для поиска данных и гетерогенного доступа к ним. Основу сервиса составляет распределенный механизм запросов, который позволяет исполнять гетерогенные SQL-запросы к различным СУБД.

Сервис управления метаданными является одним из ключевых инструментов процесса управления данными и позволяет: сформировать описания данных; повысить качество аналитики; оптимизировать подходы к хранению (ILM); обеспечить рост качества данных; реализовать возможность переиспользования данных.

Сервис управления доступом обеспечивает хранение и управление ролевыми моделями доступа к подсистемам платформы, а также обновлением данных о последней активности пользователей и единообразное хранение ролевой модели. Задача сервиса — обеспечить управляемые, прозрачные и понятные процессы получения доступов к сервисам и данным аналитической платформы.

Мониторинг предоставляет аналитику работы аналитической платформы кибербезопасности в режиме реального времени с мониторингом поведения каждого элемента на базе ретроспективного анализа, скоринговых моделей и корреляции метрик. Включает в себя инфраструктурный мониторинг, мониторинг качества данных, мониторинг статистического и синтетического контроля.

Управление поставкой данных (Feature Store) — это «магазин» фичей, интерфейс между данными и моделями ML. Feature Store объединяет практики по управлению данными, такие, как каталогизация фичей для обучения и продакшена ML.

Среда исполнения моделей — комплекс сервисов и инструментов для пакетного и потокового исполнения моделей. Включает возможность разрабатывать AI/ML-модели и обеспечивать их работу в ПРОМе, при этом избежав сложностей, связанных с настройкой инфраструктуры и DevOps конвейера.

Сервис поиска предоставляет возможность оперативного анализ данных, в том числе

для разбора инцидентов, и обеспечивает продвинутые поисковые возможности, но при этом не требует от пользователей глубокого знания SQL и понимания внутренней структуры хранилища данных.

Лаборатория данных — единая среда для работы DataScience и аналитиков с доступом к актуальным промышленным данным, инструментам анализа и разработки, а также необходимыми аппаратными ресурсами для работы с использованием технологического стека Big Data.

Анализ и визуализация данных — набор BI-инструментов для проведения бизнес-аналитики, выявления тенденций и закономерностей, формирования отчетов и дашбордов. Включает сервисы гетерогенного доступа к данным и ETL для расчета и агрегации витрин, предоставляет широкий спектр инструментов по созданию различных видов отчетов и информационных панелей.

3.3 Принципы построения платформы Кибербезопасности

Принципы построения аналитической платформы кибербезопасности — это набор ключевых правил, определяющих архитектуру решения и задающих стратегическое направление развития.

АРХИТЕКТУРНЫЕ ПРИНЦИПЫ ОБЕСПЕЧИВАЮТ СТАНДАРТИЗАЦИЮ ПОДХОДОВ К ПРОЕКТИРОВАНИЮ ТЕХНИЧЕСКИХ РЕШЕНИЙ, СООТВЕТСТВИЕ ЗАДАННОМУ УРОВНЮ НАДЕЖНОСТИ И ДОСТУПНОСТИ

Развитие архитектуры платформы с использованием подхода CSMA (Cybersecurity Mesh Architecture). В соответствии с подходом CSMA платформа реализует уровень интеллектуальной обработки событий.

Разделение технологических сервисов и прикладных продуктов. Разделение технологических сервисов, которые выполняют типовые рутинные операции, и прикладных продуктов, в которых реализуется бизнес-функциональность.

Переиспользование. Идентичная функциональность, реализованная в существующих платформенных сервисах, переиспользуется. Исключением являются случаи, когда это приводит к образованию циклических зависимостей или невозможностью технической реализации.

Выделено десять технологических сервисов, которые выполняют рутинные (циклические) задачи: загрузка, выгрузка, хранение и обработка данных, поиск, управление политиками

доступа, мониторинг, управление метаданными, предоставление доступа к хранилищам, разработка и запуск моделей, проверка гипотез в лаборатории.

Единый интеграционный слой. Платформа должна предоставлять централизованный набор сервисов для взаимодействия с потребителями и доступа к данным.

Интеграционный слой платформы кибербезопасности включает брокер сообщений (kafka), сервис доступа к данным, набор унифицированных API.

Централизованное подключение источников. Для получения событий и загрузки данных необходимо использовать инструменты интеграционного слоя, которые обеспечивают возможность переиспользования данных всеми продуктами на платформы.

Подключение всех потоковых и пакетных источников осуществляется средствами централизованного сервиса «Загрузка».

Публикация метаданных источников. При подключении новых источников обязательна публикация метаданных в единый каталог. Обязательными к описанию метаданными являются физическая модель, описание атрибутов, владелец данных, глубина хранения данных.

Используется специализированный сервис управления метаданными. Метаданные используются для организации поиска, управления доступом, выгрузки данных.

Централизованное хранение данных. Платформа должна предоставлять централизованные инструменты для хранения данных.

Реализовано оперативное и аналитическое хранилище данные, разработан сервис централизованного доступа к данным.

Доступ к данным и приложениям на основе контрактов. Совместное использование данных и функций приложений участниками должно регулироваться доступом по контракту. Только на основании заключенного контракта компонент может предоставлять доступ к своим данным и функциям.

По всем интеграциям платформы и доступу к данным заключается контракт (SLA).

Изоляция работы с данными. Требуется изолировать работу прикладной логики от физического слоя хранения данных.

Все обращения к хранилищам данных осуществляются через платформенный сервис доступа к данным, который обеспечивает обработку запроса в терминах логической модели. Пользователю нет необходимости понимать физическую модель данных, достаточно знать логическую модель данных.

Предоставление обогащенных / агрегированных данных. Платформа выступает для внешних потребителей поставщиком только обработанных, обогащенных/агрегированных данных, витрин данных, сервисов. Платформа не может быть использована в роли интеграционной площадки при передаче необработанных данных между АС.

Интеграция на основе типовых API. Платформа должна предоставлять набор API для взаимодействия с потребителями. Прямой доступ к данным недопустим. Все API платформы должны быть опубликованы в реестре API (API Management). Должна быть обеспечена обратная совместимость версий API.

Каждый компонент платформы разрабатывает набор API.

Прозрачность действий пользователя. Требуется формирование аудита (цифрового следа) действий пользователя и администратора в платформе.

Каждый компонент платформы формирует и отправляет события в централизованный сервис аудита.

Распределённый характер платформы. Компоненты приложений и наборов данных распределены между различными инфраструктурными и географическими локациями без ущерба для функционирования.

Каждой компонент платформы кибербезопасности СБЕРа построен на основе растянутого кластера в трех ЦОД (два основных и кворумный).

3.4 Использование подходов Cloud-native и Cloud-Ready

Одним из показателей оценки эффективности платформенных систем является использование ресурсов инфраструктуры и ее утилизация. Считается, что целевые показатели по этому параметру должны находиться на отметке 60% по средней утилизации, что обеспечивает достаточный запас производительности в случае пиковых нагрузок, но при этом нет простоя оборудования. Достижение таких цифр возможно в случае, если аналитическая платформа будет использовать мощности только в тех случаях, когда это действительно необходимо.

ПРОЕКТИРОВАНИЕ АРХИТЕКТУРЫ АНАЛИТИЧЕСКОЙ ПЛАТФОРМЫ КИБЕРБЕЗОПАСНОСТИ НЕОБХОДИМО ПРОВОДИТЬ С УЧЕТОМ ПОДХОДОВ CLOUD-NATIVE И CLOUD-READY

Классический подход, при котором мощности брались заранее и «про запас» уходит в прошлое. Существующий вычислительный пул позиционируется как эластичный и внешне не имеющий границ. Вычислительное облако должно обеспечить платформу возможностью получения мощности по требованию и в любой момент.

Подход Cloud Native позволяет разрабатывать масштабируемые приложения для современных динамических сред, и использовать их как в частных, так и в публичных облаках. Архитектура Cloud Native предполагает создание высокораспредленной системы на основе микросервисного подхода, т.е. контейнеров, сервисных сетей и декларативных API.

Основу Cloud Native архитектуры составляет среда исполнения, базовое инфраструктурное ПО, обеспечивающее возможность исполнения элементов логики автоматизированных систем. Среда исполнения должна быть контейнеризированной, т.е. запускающй образы контейнеров в изолированном окружении.

При использовании Cloud Native архитектуры сервисы аналитической платформы должны строиться на основе набора исполняемых технологических компонент и обладать следующими свойствами:

- Эластичность, позволяющая изменять параметры масштабирования (вверх или вниз) в режиме, приближенном к реальному времени, на основании заданных критериев и в зависимости от меняющейся нагрузки.
- Слабая связанность, отсутствие прямых зависимостей, когда изменение одного исполняемого компонента приложения сразу однозначно ведет к изменению всех исполняемых компонентов, а взаимодействие компонент осуществляется через API.
- Отказоустойчивость, готовность автоматическому самовосстановлению любого из исполняемых компонентов приложения, наличие механизмов устранения потери и рассинхронизации данных, альтернативной ветки логики исполнения в случае отказа всех экземпляров.

Подход Cloud-Ready предполагает адаптацию существующих приложений для работы в облачной среде за счет использования механизмов виртуализации. Архитектура Cloud-Ready приложения предполагает декомпозицию на отдельные компоненты (предоставляющие API для интеграции между собой), при этом, сервера приложений и другая исполняемая логика должны поддерживать исполнение на виртуальной инфраструктуре.

Выбор Cloud Native и Cloud-Ready подхода при построении аналитической платформы кибербезопасности зависит от большого количества параметров, включающих предполагаемую нагрузку на платформу, механизмы развертывания и особенности инфраструктуры заказчика. Основные отличительные признаки двух типов подходов представлены на Рисунке 5.

	Cloud-Ready	Cloud-Native
Инфраструктура	Компоненты размещаются на стандартных виртуальных машинах	Использует микросервисный подход, компоненты размещены в контейнерах
Сопровождение	Требуется специфическая ручная инсталляция и настройка	Автоматическое управление зависимостями, обновления и патчи производятся автоматически без прерывания сервиса
Масштабируемость	Горизонтальное масштабирование, специфичное для конкретного приложения	Автоматическое масштабирование
Надежность	Механизмы восстановления, специфичные для конкретного приложения	Автоматическое восстановление

Рисунок 5. *Основные признаки Cloud Native и Cloud-Ready подхода*

3.5 Обеспечение отказоустойчивости и катастрофоустойчивости платформы

Аналитическая платформа кибербезопасности является ключевым элементом защиты компании. Отказ работоспособности системы такого класса может привести к нарушениям процессов защиты инфраструктуры и антифрод, недоступности SOC и потери контроля над объектами защиты.

Отказоустойчивость — способность системы к восстановлению функционирования в случае одиночных локальных сбоев. Соглас-

но ГОСТ Р 51904-2002 отказоустойчивость определяется как свойство системы продолжать правильное выполнение функций при наличии ограниченного числа аппаратных или программных дефектов. Отказоустойчивость обеспечивается в рамках одного ЦОД средствами кластеризации и балансировки нагрузки между компонентами.

Катастрофоустойчивость (DR, Disaster Recovery) — способность системы к восстановлению функционирования в случае полного сбоя (либо долговременной недоступности) основной площадки на оборудование, расположенное на резервной площадке, в прежней конфигурации и с актуальными данными.

Классический подход к обеспечению катастрофоустойчивости платформы требует установки дополнительного КТС и увеличения ресурсов на сопровождение. Разрабатывая концепцию георезервирования, необходимо определить стратегию резервирования для каждого сервиса в зависимости от его уровня критичности и потребности в обеспечении полноценного уровня катастрофоустойчивости.

Объем оборудования для сервисов в каждом плече определяется моделью RCO (Recovery Capacity Objective) и прямо пропорционален планируемой проектной мощности, уровню критичности резервируемых сервисов и проценту снижения нагрузки при выводе из строя основной площадки.

В штатном режиме весь входящий трафик приходит на основное плечо и далее распределяется между сервисами основного и резервного плечей с помощью геобалансировщика. Stateless сервисы — кластеры k8s, кластеры серверов приложений (spring boot/wildfly) и прочие — работают в режиме active-active. В основном и резервном плечах установлен собственный набор stateless сервисов, не связанных друг с другом.

Сервисы хранения (обеспечивающие персистентное хранение данных) — СУБД, kafka, кеширующие сервисы, файловое хранилище и прочие могут работать в одном из 3 режимов:

- Несвязанные локальные кластера: один кластер в каждом плече, кластера не связаны между собой.
- Геокластер active-active.
- Геокластер active-standby.

Выделенные каналы связи используются для передачи пользовательского трафика и репликации данных сервисов хранения между плечами. Для отказоустойчивости требуется наличие двух независимых каналов для каждой пары ЦОД.

Помимо двух основных плечей рекомендуется использовать кворумный ЦОД, включающий минимальный набор компонентов, обеспечивающих кворум для геокластеров.

4. Технологические сервисы платформы

4.1 Потоковая загрузка и обработка

Сервис потоковой загрузки представляет собой набор инструментов для быстрого подключения к аналитической платформе потоковых источников данных и предназначен для online обработки данных, поступающих как поток записей или событий в online сценариях. При этом, дополнительно, данные могут быть обогащены из различных источников на основе идентификаторов, содержащихся в поступающих сообщениях, при условии, что источники обеспечивают достаточную производительность при запросе и передаче данных.

Результатом работы сервиса может быть поток новых событий, полученных на основе входных данных, поток обогащенных событий, готовая витрина. Важным преимуществом при этом является возможность обогащать данные из NRT реплики источника с минимальным отставанием и с контролем доступа.

Сервис потоковой загрузки обеспечивает:

- сбор данных в режиме реального времени из множества источников;
- валидацию и нормализацию загружаемых на потоке данных;
- предоставление данных в потоковом режиме для online-анализа;
- сохранение данных для долгосрочного хранения и дальнейшего offline-анализа.

В платформе кибербезопасности СБЕРа технологический стек сервиса потоковой загрузки построен на основе распространенных и хорошо масштабируемых open-source решений: Apache Flink, Apache Kafka, ClickHouse, Nginx, Syslog-ng, pmacct.

В качестве фреймворка для потоковой обработки данных используется Apache Flink. Для первичного сбора и хранения «сырых» потоковых данных используется Apache Kafka. Для

ЦЕННОСТЬ ДЛЯ SOC:

Сбор, нормализация и хранение всех необходимых событий КБ, данных обогащения.

Аналитический и ретроспективный поиск по всему объему данных КБ.

Мнение заказчика

хранения данных каждого источника используется отдельный топик (Рисунок 6).

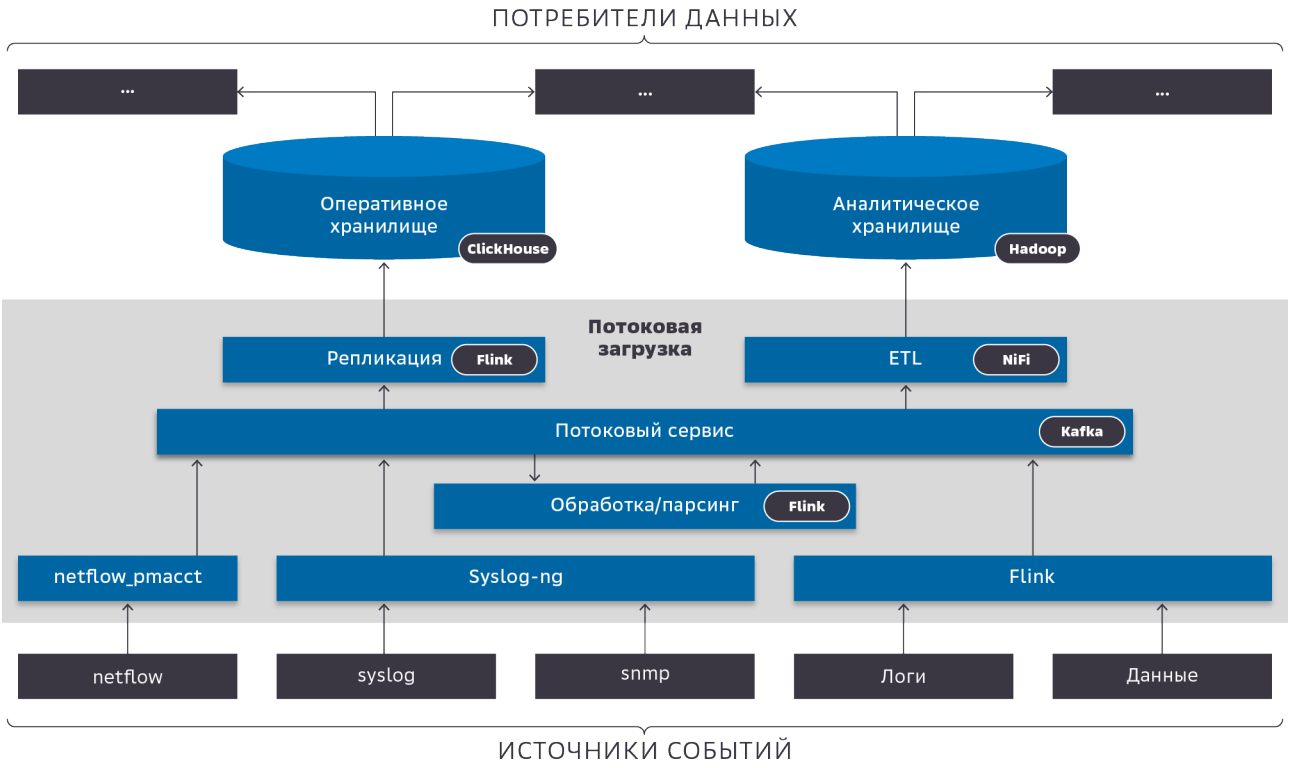


Рисунок 6. *Потоковая загрузка*

При реализации всех компонентов сервиса используется подход LowCode, при котором вся функциональность компоненты представлена в приложении-ядре, а логика работы задается с помощью метаданных. Такой подход минимизирует количество релизов, необходимых для изменения логики загрузки/обработки данных. Пример интерфейса пользователя сервиса потоковой загрузки (Рисунок 7).

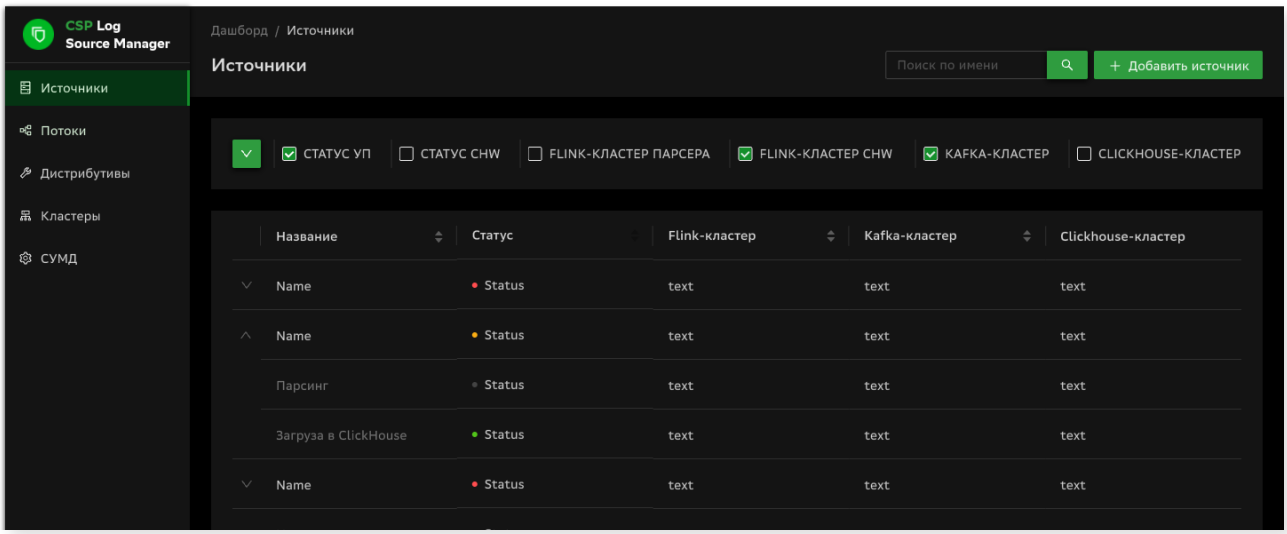


Рисунок 7. *Интерфейс потоковой загрузки (LowCode)*

После сбора сырых потоковых данных возникает необходимость приведения их к единой структуре для удобства работы с ними потоковых потребителей, т.к. сырые данные представляют собой в большинстве случаев неструктурированные инфраструктурные логи.

За эту функцию отвечает основной компонент «сервиса потоковой загрузки» — универсальный парсер. Это приложение, которое умеет обрабатывать различные форматы неструктурированных входных данных, превращая их в конечную структуру в виде JSON. Правила нормализации любого источника для универсального парсера представлены метаданными, с помощью которых определяются этапы парсинга, условия переходов между этапами и необходимые операции трансформаций/маппинга над данными на каждом этапе. Результаты нормализации в режиме онлайн загружаются обратно в Apache Kafka, где они сразу доступны потоковым потребителям для online-анализа.

Потоковая загрузка в СБЕРе:

247
потоковых источников

6+ млн.
EPS

180
ТБ/сутки

Актуальная JSON-схема источника отправляется в сервис управления метаданными для обеспечения потребителей данных актуальной физической схемой потокового источника, что упрощает его дальнейшее использование.

За агрегацию потоковых данных отвечает отдельная компонента сервиса. С помощью метаданных достаточно определить ключевые поля, определить агрегирующие функции и временное окно. Правильно подобранные параметры позволяют значительно снизить объем данных после агрегации, что позволяет сэкономить место для дальнейшего хранения данных.

Практика СБЕРа показывает, что использование хорошо зарекомендовавших себя open-source инструментов для хранения/обработки потоковых данных позволяет построить производительный, надежный и хорошо масштабируемый сервис, который позволяет обработать практически неограниченный объем потоковых данных. Выбранный принцип LowCode при реализации компонентов сервиса позволяет повысить удобство использования сервиса для пользователей.

4.2 Пакетная загрузка

Сервис пакетной загрузки предназначен для получения данных по расписанию, при этом данные загружаются batch-ами с задержкой по времени от момента их появления в системе источнике и используются для offline обработки. Процесс загрузки включает в себя трансформацию данных, парсинг, обогащение.

Пакетная загрузка данных включает набор инструментов для быстрого подключения к платформе различных источников информации. Следует отметить, что источниками данных для пакетной загрузки могут быть самые разнообразные виды информации. Например, данные из CRM и АБС систем, профили клиентов (которые могут содержать персональные данные и информацию о поведении клиентов), транзакции (в т.ч. информация о платежах и финансовых операциях), кредитные и депозитные портфели.

Пакетная загрузка данных в аналитическую платформу кибербезопасности представляет собой сложный многоэтапный процесс, позволяющий обеспечить эффективное функционирование платформы и обработку больших объемов данных. Это важный аспект, обеспечивающий безопасность и надежность работы любой платформы (Рисунок 8).

Рисунок 8. *Сервис пакетной загрузки*

Первоначальный этап заключается в быстрой доставке исходных данных к кластеру без предварительной трансформации с помощью ETL инструментов (например, Apache NiFi). На этом этапе пользователи сервиса могут применять различные шаблоны.

Сервис должен работать с широким спектром форматов данных, включая JSON, Jolt-трансформации, CSV, XML, Parquet, PDF, JPEG и др. Каждый из этих форматов требует своего подхода к обработке, включая парсинг данных, проверку подписи, валидацию и преобразование данных в табличное представление.

Источники передают данные в платформу, используя один из физических транспортов/протоколов, поддерживаемых сервисом пакетной загрузки — WebHDFS, HTTP, JMS, Apache Kafka, JDBC, файловая система.

Формат сообщения/файла также один из поддерживаемых — «произвольный файл» (не трансформируется в табличную форму, а кодируется в Base64 и целиком сохраняется в поле таблицы Hive; обязанность трансформации в этом случае ложится на конечного потребителя этих данных на выходе из ПКБ), или сообщения с атрибутами, которые можно трансформировать в табличную форму — CSV, JSON, Parquet, XML.

По результату загрузки в общем случае на одну таблицу/бизнес-сущность источника формируется несколько external-таблиц Hive (src- и inc-слои) над файлами parquet (формат является стандартом ПКБ, обоснование выбора выходит за рамки этой статьи). В location таблиц закодированы имя источника, слоя и таблицы. Имена схем в Hive также содержат имя источника и слоя. В таблицы добавляются поля с мета-информацией — номер и дата сеанса загрузки, синтетический ID файла/сообщения (в src-слое это один ID, в inc-слое два — ссылка на src-слой, и собственный ID сообщения в inc-слое).

Унифицированный, обобщённый процесс загрузки конфигурируется метаданными — набором настроек для каждого конкретного источника.

Как правило, загрузка данных включает в себя большое количество ручных операций. Необходимо заполнить метаданные в определенном формате, убедиться, что все обязательные параметры указаны, запустить сборку в Jenkins, деплой

Пакетная загрузка в СБЕРе:

143

уникальных поставщика данных

75

ТБ/сутки загружаемый инкремент

сборки на тестовый стенд и провести тестирование.

В аналитической платформе кибербезопасности СБЕРа для упрощения процесса загрузки данных активно используется подход Low Code, который позволяет пользователям с небольшим техническим опытом создавать новые механизмы загрузки, работать с зафиксированным форматом метаданных, преобразовывать данные в момент загрузки с помощью jolt, xslt. Пример интерфейса пользователя сервиса пакетной загрузки (Рисунок 9).

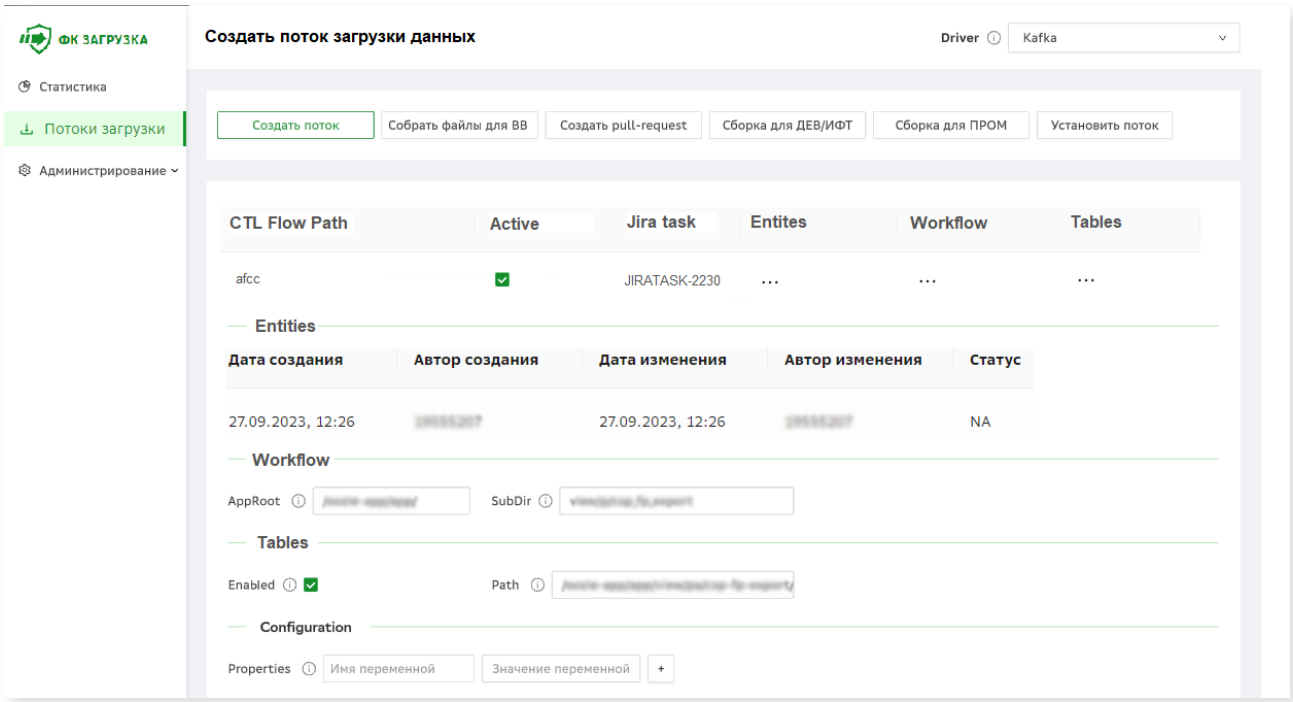


Рисунок 9. *Интерфейс пакетной загрузки (LowCode)*

4.3 Интеграционный сервис

Интеграционный сервис обеспечивает информационное взаимодействие между компонентами платформы и сторонними продуктами. Основу сервиса составляют продукты на базе решений с открытым исходным кодом Apache Kafka и приложения в спецификации OpenAPI.

Apache Kafka включает распределенное хранилище сообщений (чаще всего логов) и key-value базы данных. В платформе кибербезопасности эта распределенная платформа потоковой передачи событий используется для обмена сообщениями при интеграции различных компонентов платформы (как

продуктов, так и технологических сервисов) между собой.

Apache Kafka реализует принцип «издатель — подписчик», т.е. источники записывают сообщения в топик, а потребители (подписанные на данный топик) считывают эти события в режиме реального времени. Топик — это логическое хранилище сообщений для потребителей, который позволяет сгруппировать потоки сообщений по категориям. Топик может быть разбит на разделы, а раздел представляет собой журнал сообщений от одного источника.

Практика СБЕРа показала, что настройки Kafka варьируются в зависимости от объема и структуры поступающих событий. Для оптимальной работы Kafka необходим индивидуальный подход к настройкам каждого топика и раздела. В случаях большого объема событий потоки необходимо разделять по разным кластерам Kafka, группируя схожие по формату источники.

OpenAPI — формализованная спецификация для описания API, которая представлена в формате JSON или YAML и содержит подробное описание методов, параметров, структуры данных и другой информации, необходимой для взаимодействия с платформой. При создании API необходимо придерживаться следующих принципов: фиксация контракта, возможность повторного использования, выполнение одного действия над одной сущностью.

Контракт API (SLA) — фиксирует ответственность поставщика и ожидания потребителя и включает как функциональную составляющую (сигнатура метода, семантика), так и целевые нефункциональные параметры. В контракте API фиксируются:

- Наименование API;
- Версия API;
- Функциональные параметры (входные данные, возвращаемый результат, описание функциональности);
- Нефункциональные параметры (доступность, время ответа, пропускная способность, технологические ограничения, ограничения безопасности) – целевые значения, исходя из которых разрабатывается API.

Дальнейшее изменение контракта возможно только через выпуск новой версии API и согласование измененной специ-

фикации со всеми потребителями.

Активное развитие платформы, создание новых и доработка существующих компонентов/модулей привносит многочисленные изменения в ранее разработанный код. В таких условиях требуется управлять вносимыми изменениями, для этого вводится понятие — версионность API. Версионность API позволяет решать проблемы обратной совместимости, снизить риски возникновения регресса на поздних стадиях тестирования и вывода в промышленную эксплуатацию, повысить надежность и стабильность реализуемых компонентов/модулей.

При построении интеграционных взаимодействий через API основным критерием является совместимость версий. Изменение версии API платформы не должно влиять на потребителей.

4.4 Организация хранения данных

Достаточно длительное время данные, необходимые для аналитической поддержки продуктов кибербезопасности, группировались вокруг существующих процессов и решений. Аналитические платформы кибербезопасности позволили перейти на дата-центричный подход, в котором главным звеном являются данные, вокруг которых строятся платформенные продукты и процессы. Одной из важных задач при построении платформы являлась организация сбора, хранения и обработки данных из множества источников.

Для оптимизации хранения данных используются оперативное и аналитическое хранилища.

- **Оперативное хранилище** (ClickHouse) — хранение данных для выполнения «на лету» простых (без join) поисковых и агрегационных запросов.
- **Аналитическое хранилище** (Sberbank Data Platform, SDP Hadoop) — длительное хранение структурированных и неструктурированных данных для выполнение различных аналитических расчётов, обучения моделей, а также в качестве архива.

4.4.1 Оперативное хранилище

ClickHouse — это столбцовая система управления базами данных (СУБД) для онлайн обработки аналитических запросов, удовлетворяет требованиям по надежности, масштабируемости и функциональности. Основные характеристики оперативного хранилища:

- подавляющее большинство запросов — на чтение;
- данные обновляются достаточно большими пачками (> 1000 строк), а не по одной строке или не обновляются вообще;
- при чтении вынимается достаточно большое количество строк из БД, но только небольшое подмножество столбцов;
- таблицы содержат большое количество столбцов;
- высокая нагрузка при обработке одного запроса (до миллиардов строк в секунду на один сервер);
- результаты выполнения запроса, как правило, существенно меньше исходных данных.

Важной особенностью ClickHouse является то, что он спроектирован для работы на обычных жестких дисках, что обеспечивает низкую стоимость хранения. Данные в ClickHouse могут быть расположены на разных шардах. Каждый шард может представлять собой группу реплик, которые ис-

пользуются для отказоустойчивости. Запрос будет выполнен на всех шардах параллельно, что увеличивает скорость построения нужных витрин. Гибкий подход в организации кластера позволяет его сделать более быстрым и отказоустойчивым (Рисунок 10).

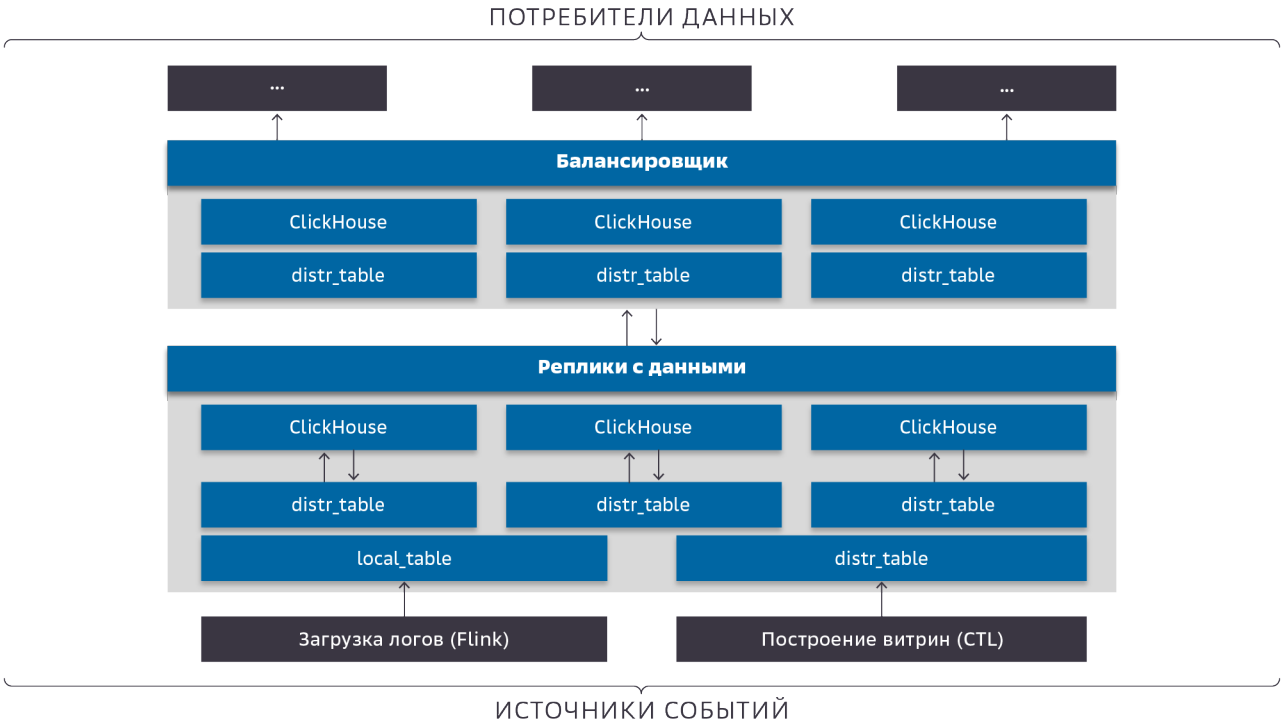


Рисунок 10. *Оперативное хранилище*

SBERBANK DATA PLATFORM (SDP HADOOP) — СОБСТВЕННАЯ СБОРКА ФРЕЙМВОРКА ИЗ КОМПОНЕНТОВ ЭКОСИСТЕМЫ APACHE HADOOP, КОТОРАЯ ПОЗВОЛЯЕТ ГИБКО И БЕЗОПАСНО РЕШАТЬ ЗАДАЧИ ОБРАБОТКИ, ХРАНЕНИЯ И АНАЛИЗА БОЛЬШИХ ОБЪЕМОВ ДАННЫХ

4.4.2 Аналитическое хранилище

Для долгосрочного хранения данных и проведения аналитических преобразований в СБЕРе используется Sberbank Data Platform (SDP Hadoop), который используется для распределенного хранения больших объемов информации; распределенных вычислений; управления и планирования вычислительных ресурсов.

Данные большинства систем источников загружаются в аналитическое хранилище в сыром виде как их предоставляет система-источник (файлы, json, syslog). далее происходит переформатирование исходных данных в унифицированный формат хранения — parquet файлы. Процессы прозагрузки данных по различным слоям, а также формирование производных витрин производится с помощью Apache Spark под управлением оркестратора «Управление оффлайн вычислениями (CTL)».

Функции:

- Хранение и обработка загруженных данных
- Создание производных витрин данных
- Подготовка данных для внешних потребителей
- Подготовка данных для аналитической отчётности
- Обучение моделей
- Архивное хранение данных

В хранилище выделяют различные слои хранения данных согласно классическим подходам построения хранилищ данных (Рисунок 11):

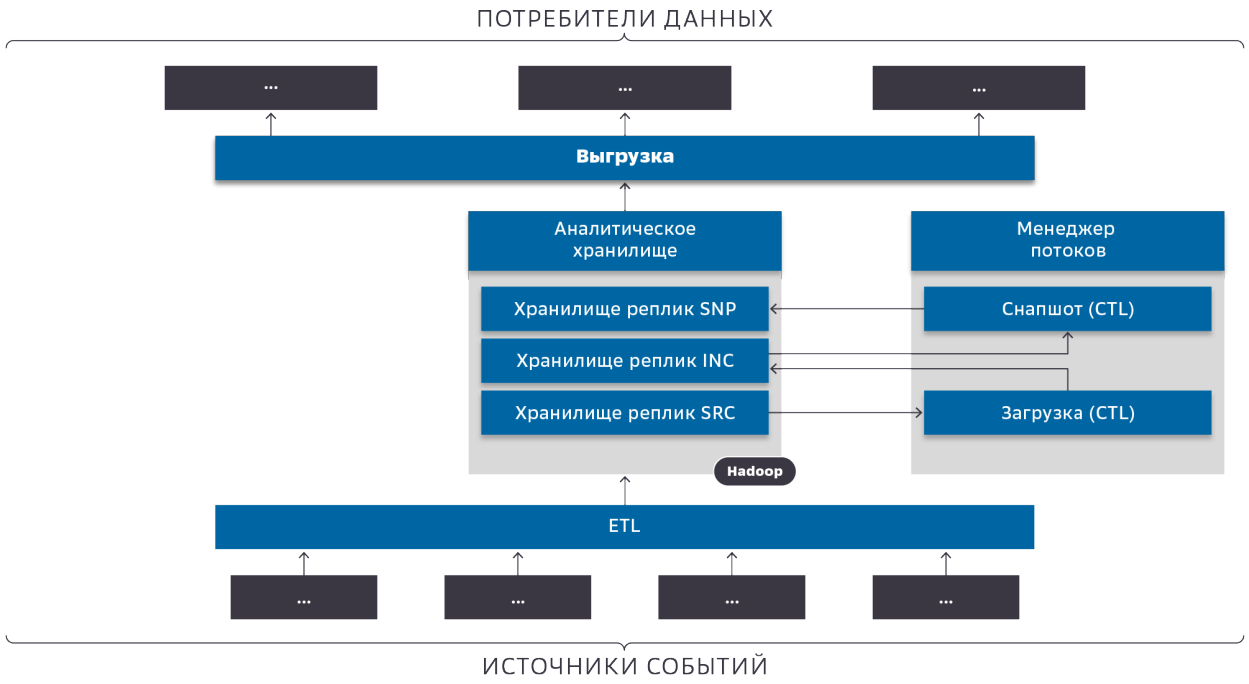


Рисунок 11. *Аналитическое хранилище*

- **src** – хранение исходных данных систем-источников;
- **inc** – инкремент от систем-источников в формате parquet;
- **snp** – сформированный из инкрементов снэпшот данных. По сути – слепок данных системы источника на момент выгрузки данных;
- **hist** – исторический снэпшот;
- **dm** – витрины данных;
- **exp** – данные для выгрузки во внешние системы.

Аналитическое хранилище по сути является хабом данных для последующего использования в различных продуктах платформы. Команды каждого продукта могут создавать производные от данных систем-источников – различные ви-

трины данных, агрегаты, а также выгружать их в продуктовые хранилища или другие автоматизированные системы.

С использованием лаборатории данных команды могут выполнять различные Ad-Hoc запросы к данным и проводить обучение различных моделей данных.

4.4.3 Политики управления данными

Политика управления данными (Data Policy) — это набор правил, которые помогают защитить данные и устанавливают стандарты для их доступа, использования и целостности.

Владельцы данных. Все данные системы должны иметь владельца. Владелец отвечает за сами данные (качество, целостность, актуальность), согласует права доступа к данным, отвечает за политики хранения данных.

Наличие владельца для каждого объекта позволяет оперативно отвечать на вопросы мониторинга, аналитики по данным, а также проводить «аллокацию стоимости данных» на владельцев.

Классификация данных. Для управления данными необходимо проводить их классификацию на различные типы хранимой информации. Исходя из классификации данных и понимания сценариев их использования выстраиваются дальнейшие процессы управления этими данными. При этом классификация данных может различаться в различных хранилищах и продуктах.

- Пример классификации данных для Хранилища на базе Hadoop.
- Сырые данные источников (Source). Данные, приходящие от систем-источников в любом виде.
- Временные данные (Temp). Технические данные, используемые в работе процессов и пользователей.
- Промежуточные данные (Stage). Данные процессов, используемые для промежуточных расчётов. Зачастую используются для отладки и анализа инцидентов.
- Аналитические данные (Analytics). Структурированные данные, на которых строится аналитика, отчетность.

Управление данными в хранилищах:

Компрессия. Сжатие данных требует дополнительного времени обработки при записи и чтении файлов, но при этом даёт выигрыш по занимаемому месту до 10 раз. Также, в итоге, за счёт обработки меньшего объёма информации, получаем выигрыш по производительности.

Архивирование. Процесс переноса данных на более дешевые носители вследствие их устаревания и уменьшения использования.

Логически данные можно разделить на:

- горячие — используемые очень часто в текущий момент времени (пример — данные по продажам за текущий месяц);
- тёплые — данные, используемые для аналитики, а также при закрытии каких-либо периодов (пример — данные по продажам за текущий год);
- холодные — данные, изредка используемые по запросу или для исторической аналитики (пример — данные по продажам старше 5 лет).

Например, самые «горячие» данные хранят в памяти (в том числе In memory DB), в базах данных на Hi-End, MId-Range оборудовании. Архивные или другими словами «холодные» данные сохраняются в HDFS, на ленту и на другие дешёвые носители.

Снижение дублируемости данных. Для

обеспечения отказоустойчивости данные зачастую дублируются — репликация в кластерных решениях (троируют и т.д.). К примеру, в HDFS по умолчанию устанавливается фактор репликации — 3 (3 копии), такая же практика используется в других кластерных решениях. В случае временных данных, а также данных «на всякий случай» допускается уменьшение дублируемости данных, не влияющее на работоспособность каких-либо процессов.

Глубина хранения. Для данных, которые никак не используются, или стоимость их хранения несоизмерима с эффектом от использования, устанавливается глубина хранения, по истечении которого данные полностью удаляются. Глубину хранения данных объекта определяет их владелец.

4.5 Сервис доступа к данным

Подходы к обработке информации, размещаемой в хранилищах платформы, очень сильно разнятся в зависимости от природы и назначения этой информации. Так, например, информация для мгновенного реагирования и выявления угроз должна быстро загружаться, а поиск по ней должен быть максимально быстрым с точки зрения выгрузки небольших порций данных и агрегатов по ним. Информация, предназначенная для обучения моделей, напротив, требует глубокой и «тяжелой» переработки огромных массивов информации. В связи с этим «серебряной пули», в части выбора технологии хранения, не существует. Все это приводит к большому количеству различного рода СУБД и прочих хранилищ информации.

По мере роста количества программных решений, использующих Платформу, повышается и вариативность задач, решаемых этими программными средствами. Практика показывает, что далеко не всегда информация из конкретного хранилища будет использоваться для решения задач, под которое это хранилище подходит лучше всего. Или же данные в разных хранилищах могут иметь схожую природу, и требуется их совокупная обработка.

При таких условиях каждому программному решению, использующему Платформу, необходимо, в худшем случае, иметь количество коннекторов к системам хранения, совпадающее

СТАНОВИТСЯ ОЧЕВИДНОЙ ПОТРЕБНОСТЬ В ПРОГРАММНОМ СРЕДСТВЕ, ПОЗВОЛЯЮЩЕМ ИНТЕГРИРОВАТЬ В КОНТУР ПЛАТФОРМЫ ЛЮБЫЕ ВНЕШНИЕ ИСТОЧНИКИ И ХРАНИЛИЩА И ТИРАЖИРУЮЩЕМ ЭТИ ДАННЫЕ ПО ВСЕМ ПРОГРАММНЫМ РЕШЕНИЯМ ПЛАТФОРМЫ В РЕЖИМЕ ОДНОГО ОКНА

с их количеством, а любую логику обработки данных из смежных хранилищ необходимо реализовать на самом сервисе.

Добавив к этому необходимость централизованного контроля за доступом к данным всех систем хранения, а также наличие различных интеграций с другими программными средствами в контуре или за пределами Платформы, вырисовывается довольно серьезная проблема в этой части.

Данное программное средство должно удовлетворять следующим требованиям:

- разграничение доступа на всех уровнях от источника целиком до отдельных сущностей;
- полная наблюдаемость за всеми процессами доступа к данным;
- единый интерфейс доступа к данным, абстрагированный от характера и структуры источника (гетерогенность);
- возможность предобработки данных, в том числе из нескольких источников одновременно;
- ограничение потребителей по количеству используемых ресурсов.

В качестве такого решения на Платформе используется Сервис доступа к данным (СДД). Ядровым элементом сервиса является OpenSource решение Trino. Также в составе сервиса присутствуют микросервисы обработки запросов (синхронный и асинхронный), динамического конфигурирования и выполнения регулярных задач (Рисунок 12).

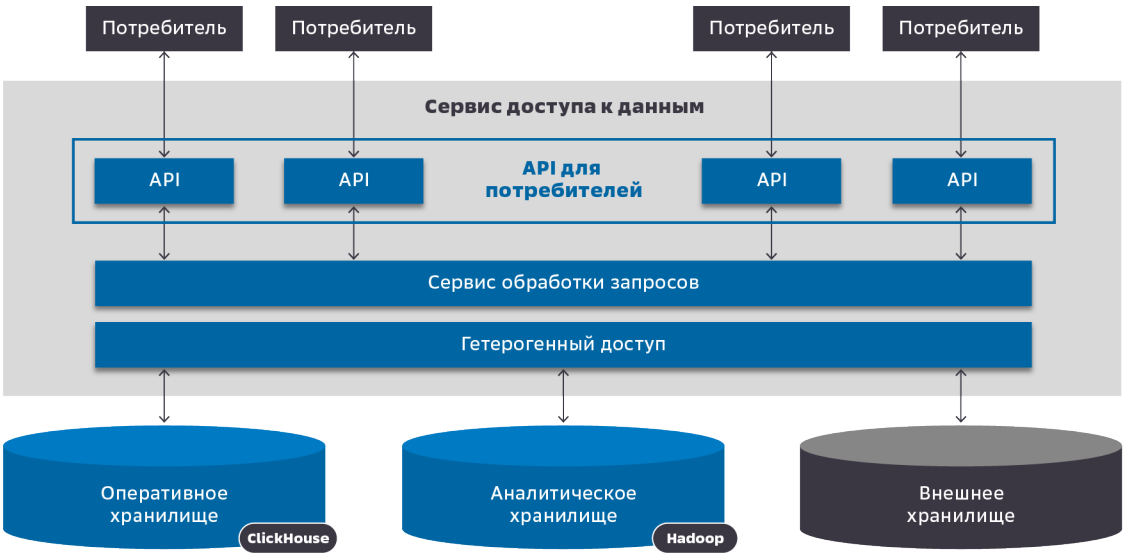


Рисунок 12. *Сервис доступа к данным*

Сервис работает по протоколу HTTP с использованием mTLS-соединения, в том числе для идентификации и аутентификации. Базовый режим работы сервиса — асинхронный. Это обусловлено тем, что в силу больших объемов хранилищ и записей в них запросы могут выполняться минуты, часы, а иногда и несколько дней (Рисунок 13).

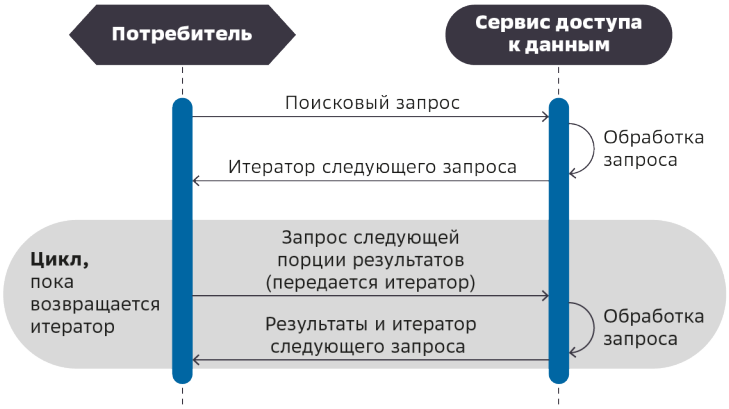


Рисунок 13. *Обработка поискового запроса*

Одновременно с этим некоторые готовые решения, которые необходимо интегрировать с сервисом доступа к данным, не имеют возможности или имеют ограниченные возможности для работы по такой схеме. Для этого Сервис имеет отдельный микросервис, обеспечивающий синхронный режим доступа к данным (фактически он является аккумулятором, позволяющим накопить и отдать единым датасетом все результаты выполнения асинхронного запроса).

В части разграничения уровней доступа к данным сервис позволяет гибко настраивать набор источников и сущностей, к которым тот или иной потребитель имеет права доступа.

В части контроля и надежности сервис поддержит полный набор инструментов наблюдаемости: логирование, аудит, мониторинг.

Сервис доступа к данным должен предоставлять следующие базовые режимы доступа:

- запрос в синтаксисе SQL;
- запрос в нативном синтаксисе хранилища;
- шаблонизированный запрос.

Запрос в синтаксисе SQL. В качестве единого синтаксиса может использовать синтаксис ANSI-SQL диалекта Trino.

В случае написания таких запросов возможно производить гетерогенные SQL-запросы к нескольким хранилищам одновременно:

```
SELECT * FROM storage_1.db.table as s1 JOIN storage_2.db.table as s2 WHERE s1.Id = s2.Id
```

где storage_1 и storage_2 – разные источники с разными системами хранения данных.

Запрос в нативном синтаксисе хранилища. Зачастую при работе с BigData-хранилищами синтаксиса SQL-диалекта Trino недостаточно. Такого рода хранилища имеют свои особенные функции и операторы, которые позволяют наиболее быстро и эффективно выполнять сложные запросы. Для этого в рамках сервиса используются дополнительные синтаксические структуры («синтаксический сахар»):

```
SELECT * FROM storage_1.db.table as s1 JOIN storage_2.NativeQuery('Select SpecificFunc(id) as Id FROM db.table') as s2 WHERE s1.Id = s2.Id
```

в данном случае подзапрос будет выполнен в нативном синтаксисе источника storage_2 и произведена операция JOIN результатов выполнения этого подзапроса с другим источником storage_1.

Шаблонизированный запрос. Данный режим доступа к данным ориентирован в первую очередь на потребителей за пределами контура Платформы и представляет собой аналог вызова удаленной процедуры, когда сам текст SQL-запроса хранится в СДД, а потребитель просто использует шаблон с указанием допустимых параметров для подстановки в этот шаблон.

Сложные виды доступа к данным. Наличие простых и гибких механизмов доступа к данным, которые рассматривались выше, позволяет потребителю решать практически любые задачи. Если рассматривать возможность отправлять SQL-запросы, как рост функционала доступа к данным «в высоту», то зачастую возникает довольно большой объем задач «в ширину», когда необходимо осуществлять поиск по множеству схожих источников, сущностей и их атрибутам. Для этого СДД предоставляет ряд инструментов, позволяющий «широковещательно» опросить практически неограниченное коли-

чество источников и сущностей:

- сквозной поиск;
- логический поиск;
- поиск по вхождению.

Сквозной поиск. Представим задачу, что имеем множество сущностей, имеющих схожую природу и состав полей, но разный бизнес-смысл и сущности хранилища, в которых они размещаются (например, инфраструктурные и прикладные логи различных программных и аппаратно-программных систем). При проведении расследований инцидентов кибербезопасности зачастую требуется поиск всех событий, обладающих заданным признаком, например, требуется получить все логи со всех систем, у которых IP-адрес инициатора взаимодействия равен 192.168.0.1. Если сущности с логами составляют десятки или сотни единиц — то формирование такого рода запроса будет довольно затруднительным. Для решения этой задачи был реализован механизм доступа к данным «Сквозной поиск». Допустим, что во всех сущностях с логами есть колонка/поле с названием «SourceIP». В этом случае потребителю вместо составления объемного SQL-запроса требуется отправить один HTTP-запрос следующего вида:

```
{
  «selectColumnList»: [«Id «, «Name», «EventID»],
  «constraints»: [{
        «columnName»: « SourceIP»,
        «operation»: «Equal»,
        «value»: «192.168.0.1»
     }
  ],
  «timeFrom»: «2023-02-16»,
  «timeTo»: «2024-02-17»
}
```

Результатом выполнения данного запроса будет датасет, содержащий указанные в поле «selectColumnList» атрибуты, а также поле «source» с указанием источника и сущности, где данная запись найдена.

Логический поиск. Расширяя парадигму сквозного поиска, можно формировать поисковые запросы не только с привязкой к физической модели источников и сущностей, но и задавая критерии поиска с привязкой к их логическому смыслу. Например — найти все события инфраструктурных логов, в которых встречается любой IP-адрес со значением 192.168.0.1. Для этого используется логическая модель данных, предоставляемая СУМД (см. раздел «Сервис управления метаданными»).

```
{
   «selectColumnList»: [«Id «, «Name», «EventID»],
    «constraints»: [
   {
    «attributeName»: « ip_address»,
    «operation»: «Equal»,
    «value»: «192.168.0.1»
   }
 ],
   «timeFrom»: «2023-02-16»,
   «timeTo»: «2024-02-17»
}
```

Поиск по вхождению. Наиболее общим видом сложного доступа к данным является поиск по вхождению – поиск по любому атрибуту любой сущности любого источника. Например, найти уникальный идентификатор во всех источниках. Пример такого запроса:

```
{
«selectColumnList»: [«Id «, «Name», «EventID»],
«searchString»: «unique_identifier»,
   «timeFrom»: «2023-02-16»,
   «timeTo»: «2024-02-17»
}
```

Очевидно, что если решать задачи по сложным видам доступа к данным «в лоб», то перебор всех таблиц и всех столбцов (в случаях поиска по вхождению) будет занимать продолжи-

тельное время. Стандартные механизмы индексации отдельных таблиц хранилищ дадут определенный прирост производительности, но скорость отдачи данных все также будет оставлять желать лучшего.

Для ускорения поиска в платформе кибербезопасности СБЕРа были апробированы и внедрены подходы, при которых с использованием материализованных представлений создаются индексные структуры, позволяющие осуществлять индексирование и поиск по произвольным полям произвольных таблиц хранилища одномоментно. Это серьезно увеличивает скорость доступа к произвольным данным.

Аналогичным образом были разработаны механизмы выделения заданных шаблоном последовательностей, а также отдельных слов. Это позволяет эффективно проиндексировать отдельные цифробуквенные последовательности, а также исключить при поиске по вхождению такого рода последовательностей конструкции вида LIKE. Это, в свою очередь, исключает полное сканирование длинных строковых полей и, как следствие, кратно увеличивает скорость поиска таких последовательностей.

В заключении хотелось бы отметить, что, реализовав все перечисленные выше механизмы, сервис управления доступом имеет разнообразный набор инструментов, позволяющий решать практически любые задачи доступа к разнородным данным, обеспечивая при этом необходимый уровень контроля и разграничения доступа потребителя к разного рода информации.

4.6 Сервис управления доступом

Сервис управления доступом обеспечивает хранение и управление ролевыми моделями доступа к подсистемам платформы, а также обновлением данных о последней активности пользователей и единообразное хранение ролевой модели.

Основной задачей СУД является обеспечение управляемых, прозрачных и понятных процессов получения доступов к сервисам и данным аналитической платформы. СУД можно сравнить с системой «умного дома», который предоставляет удобный графический интерфейс для владельцев сервисов и данных, чтобы с минимальными усилиями определять и из-

менять политики доступа к своим ресурсам. При правильном заполнении метамодели СУД доступы к соответствующим компонентам сервиса назначаются автоматически в соответствии с изменениями в ролевой модели.

Архитектура СУД состоит из следующих компонент: ядро СУД, шлюз аутентификации, модуль управления политиками, сервис аутентификации (Рисунок 14).

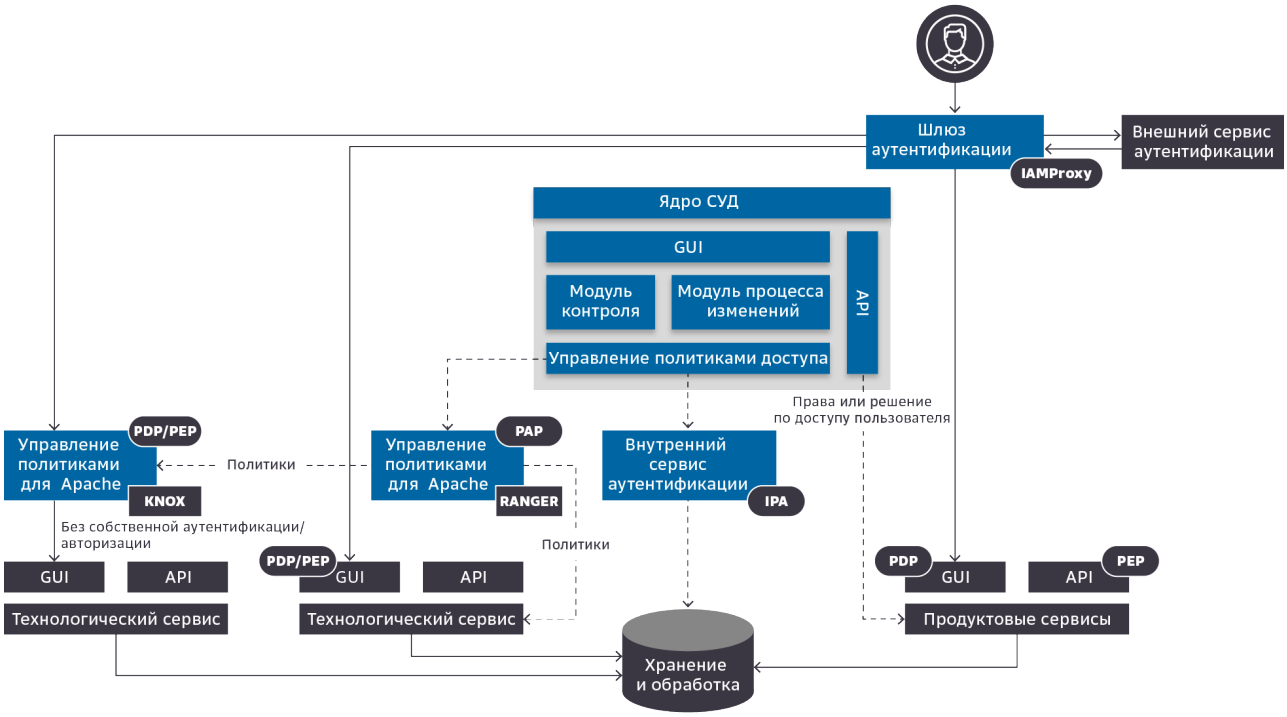


Рисунок 14. *Сервис управления доступом*

Ядро СУД — осуществляет управление и контроль правил авторизации на всех компонентах платформы и обеспечивает: управление ролевыми моделями для технологических сервисов и продуктов кибербезопасности; формирование политик доступа на основе ролевых моделей; управление учетными записям; контроль соответствия эталонных ролевых моделей и текущих политик доступа.

Модуль управления политиками безопасности продуктов Apache — хранит все политики безопасности, полученные от ядра СУД, и передает их на соответствующие компоненты технологических сервисов, использующих продукты Apache, для осуществления ими разграничения доступа пользователей и/или систем. Построен на основе продукта Apache Ranger, который ведет централизованные политики доступа для таких

сервисов как Hadoop, Hive, Nifi и т.д.

Внутренний сервис аутентификации – это служба каталогов, предназначенная для создания в ОС Linux среды, позволяющей централизованно управлять аутентификацией пользователей на базе протоколов LDAP и Kerberos. Сервис реализован с использованием продукта freeIPA.

Шлюз аутентификации пользователей – проводит аутентификацию пользователей, выступая единой точкой доступа пользователей к UI и/или API сервисов платформы. Сервис реализован с использованием продукта IAMProxy.

Шлюз авторизации пользователей – используется для разграничения доступа пользователей в тех решениях, где отсутствуют собственные механизмы авторизации. Любой пользователь, обращающийся к незащищенным сервисам, проходит через данный шлюз, который проверяет, можно ли тому обращаться к сервсису или нет на основе политики доступа. Сервис реализован с использованием продукта Apache KNOX.

Каждая крупная организация, как правило, имеет собственный сервис аутентификации для пользователей ее АС. Аутентификация пользователей платформы может осуществляться этим внешним сервисом. Роли, заведенные в СУД, передаются во внешний сервис аутентификации, там должно осуществляться назначение их пользователям. Процесс назначения ролей пользователям является внешним для СУД и определяется утвержденной ролевой моделью компании.

Внешний сервис аутентификации должен поддерживать протокол OpenID Connect на базе oAuth и протокол SCIM. В случае его отсутствия решение по аутентификации возможно реализовать на основе open source решения. Одним из вариантов такого продукта является свободно распространяемый Keycloack.

В процессе аутентификации пользователь обращается в UI продуктового или технологического сервиса, затем перенаправляется на шлюз аутентификации, который аутентифицирует пользователя по протоколу OpenID Connect. Аутентифицированный пользователь возвращается в UI сервиса. Использование mTLS между UI сервиса и шлюзом аутентификации гарантирует, что пользователь не сможет обратиться к сервису без использования шлюза аутентификации.

Для авторизации пользователей требуется ролевая модель, которая формируется для каждого сервиса платформы отдельно. Ролевая модель может быть изменена как при помощи API, так и при помощи UI. Ролевая модель состоит из ролей, которые включают в себя набор привилегий, которые в свою очередь являются определением прав доступа к ресурсу.

Сервис управления доступом оперирует следующими понятиями:

- Resource (ресурс) – это сущность, к которой разграничивается доступ. Например, это может быть endpoint, web-страница, каталог HDFS, файл, схем, таблица, колонка в БД Hive и т.п.
- Scope (права доступа) – действия, которые могут быть произведены с ресурсом и которые ограничиваются политикой доступа.
- Privilege (привилегия) = {resouce: action in scope} – определяет, какие действия над ресурсом разрешаются в рамках данной привилегии. В привилегии может быть указан только один ресурс и несколько дей-

ствий из скоупа.

- Role (роль) — набор привилегий, которые назначаются пользователю в рамках данной роли.

Управление ролевой моделью сервисов возможно через два интерфейса: API, в рамках которого сервис может передать скоуп, ресурсы, привилегии и роли; UI, в рамках которого представитель сервиса или администратор СУД может осуществлять корректировку ролевой модели (Рисунок 15).

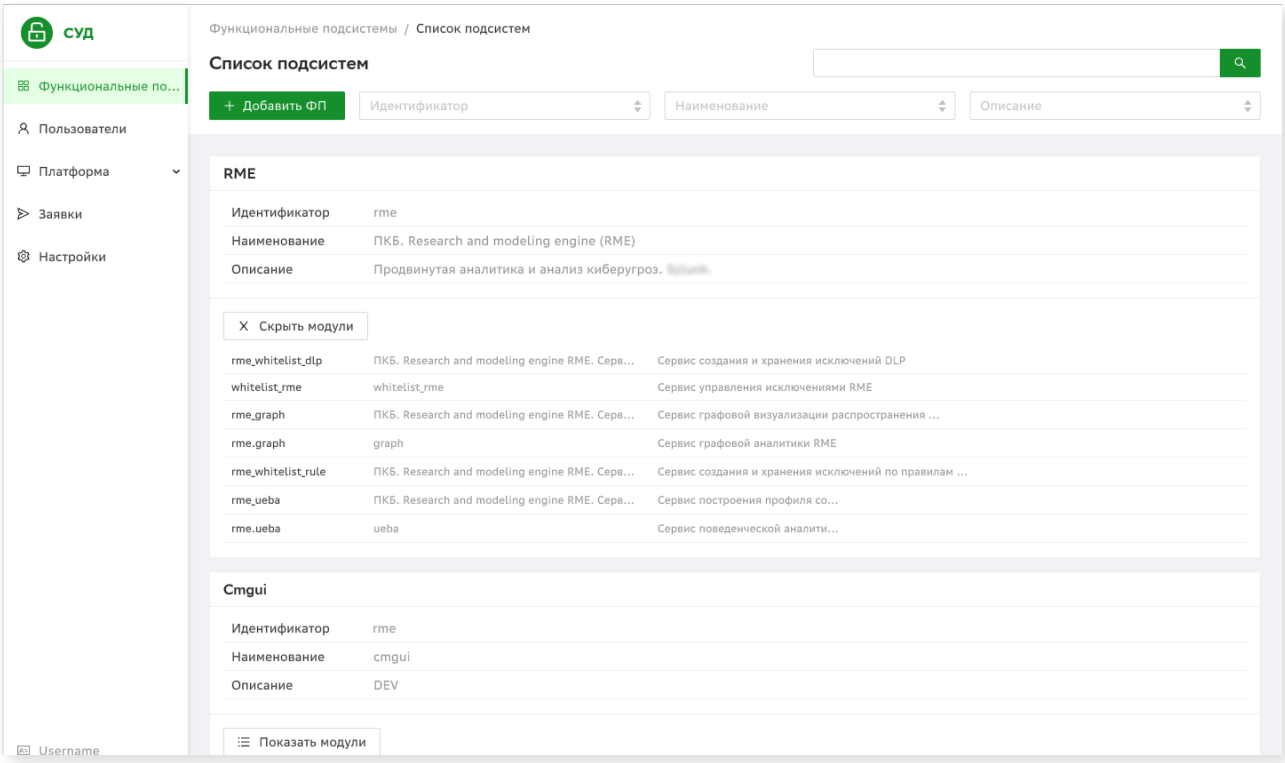


Рисунок 15. *Интерфейс Сервиса управления доступом*

Сервис управления доступом спроектирован таким образом, что часть артефактов, например, ресурсы и привилегии, можно передать через API СУД, а другие — через UI, например, роли. Любой из каналов изменения ролевой модели может быть заблокирован.

Изменение ролевой модели через UI проходит через wokrflow согласования, а цифровые следы по этому изменению записываются в аудит платформы. Для работы с изменениями ролевой модели в UI используется модуль процесса изменения. Он управляет процессами (workflow) согласования изменений, например, у нас в СБЕРе такие изменения обычно согласуются владельцем сервиса, владельцем данных и экспертом

кибербезопасности. Владельцы сервисов описаны в самом СУД, эксперты кибербезопасности определяются ролевой моделью СУД и соответствующих сервисов, а владельцы данных технологических сервисов получаются из СУМД.

Также нужно обратить внимание, что технологические сервисы не умеют взаимодействовать с API СУД, поэтому описание всех ресурсов этих сервисов автоматически подтягиваются из СУМД, а scope — из Apache Ranger. На основании ролевой модели технологического сервиса СУД формирует политику доступа и передает ее в Apache Ranger, который далее уже записывает ее на все технологические сервисы.

Если посмотреть на процесс получения доступа пользователя к ресурсу (авторизации), то в случае продуктового сервиса используется запрос к API СУД, посредством которого определяется, имеет ли пользователь доступ. При этом API спроектирован таким образом, что сервис может к нему прийти с вопросом: «Этот пользователь имеет такое-то право к такому-то ресурсу?» (в этом случае СУД выступает PDP), а может спросить «Какие у этого пользователя роли?» (в этом случае PDP будет выступать сам сервис, т.е. производить вычисление, есть ли у пользователя доступ к ресурсу или нет). Вне зависимости от того, где происходит вычисление доступа, применение решения по доступу осуществляет сам сервис (выступает в качестве PEP).

В случае технологического сервиса как вычисление (PDP), так и применение решения по доступу (PEP) осуществляет сам технологический сервис на основе политики безопасности, переданной в сервис из Apache Ranger.

4.7 Управление поставкой данных и исполнением моделей (Feature Store)

Управление поставкой данных и исполнением моделей (Feature Store) — комплекс инструментов, обеспечивающий формирование онлайн и оффлайн фичей для машинного обучения и оркестрацию исполнения on-line и off-line моделей.

Feature Store — это «магазин» фичей, интерфейс между данными и моделями ML, который объединяет механизмы по управлению данными, такие, как каталогизацию фичей для обучения и продакшена ML.

ЦЕННОСТЬ ДЛЯ АНТИФРОД:

Создавать и обучать уникальные модели выявления мошенничества, а также повысить их точность.

Проводить комплексные расследования сложных кейсов фрода.

Мнение заказчика

Feature Store позволяет формировать витрины, т.е. таблицы, являющиеся результатом агрегации исходных данных (других таблиц) в ходе выполнения над ними операций обработки данных с помощью различных инструментов (SQL-скрипты, программный код и т.д.). Результатом агрегации также может быть фича, а таблица, состоящая из фичей, будет витриной фичей.

Фича (feature) – агрегат, построенный на основе исходной выборки данных по определенному ключу (например, по идентификатору пользователя или устройства). Агрегат представляет собой значение, которое было получено в результате применения математической функции к набору исходных данных.

Использование инструментов Feature Store обусловлено следующими причинами:

- модели становятся сложнее, больше параметров может использоваться для обучения;
- модели учатся на фичах (специально подготовленных агрегатах данных, которые характеризуют некий признак объекта или процесса).

Фичи формируются специалистами data science для решения конкретных задач. В итоге множество фичей создаются разными дата инженерами для решения схожих задач, характеризуют одно и то же, строятся вокруг одного и того же ключа, но при этом не переиспользуются. В результате возникает переполнение хранилища однотипными данными.

FEATURE STORE ВКЛЮЧАЕТ РЕЕСТР ФИЧЕЙ И ОБЕСПЕЧИВАЕТ ИХ ПЕРЕИСПОЛЬЗОВАНИЕ МЕЖДУ МОДЕЛЯМИ

Задача сервиса управления поставкой данных и исполнением моделей (Feature Store) создать реестр фичей и обеспечить их переиспользование между моделями. Feature Store включает механизм для расчёта и управления фичами, т.е. конструктор, обеспечивающий их построение и управление настройками. В результате в Feature Store хранится вся информация о фичах: исходный код; описание, связи (Рисунок 16).

Функциональные возможности Feature Store:

- **Режим расчета BackFilling** – это возможность посчитать витрину, «вернувшись в прошлое», то есть создать такие условия расчета по данным, которые были на кластере на выбранный момент в прошлом. Это позволяет обучать модели на основе имеющихся исторических данных.

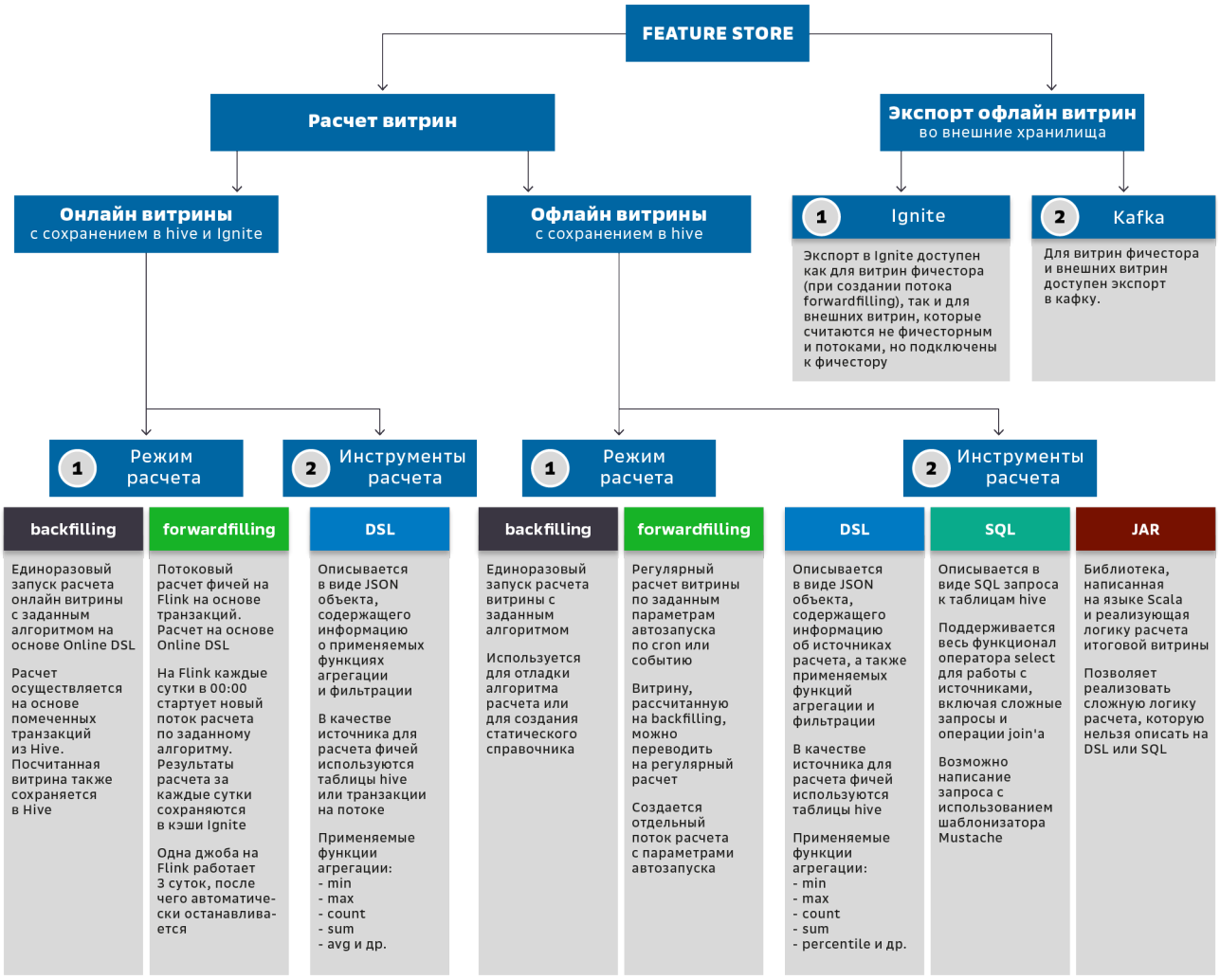


Рисунок 16. *Feature Store*

- **Режим расчета ForwardFilling** — это возможность запустить регулярный расчет витрины, опираясь на обычное календарное расписание или используя событийную модель срабатывания.
- **Версионирование фичей** — наблюдение за версиями и логикой расчета фичей на исторической прямой. Если нужно ввести новую переменную, можно посмотреть, как она влияет на точность фичей. И «зашить» ее в логику расчетов таким образом, чтобы качество модели не пострадало. Таким образом, эволюция версий модели происходит, не ломая старые версии. Чем достигается безопасность работы текущих бизнес-процессов, при внедрении новых версий моделей.
- **Офлайн-расчеты** — возможность расчета витрины в офлайн-режиме. Можно организовать на базе простого

хранилища данных (DWH). Поддержка on-premise — реализация всего пайплайна внутри собственной IT-инфраструктуры. Особенно важно для тех, кто не доверяет публичным облакам свои фичи и «секретные статистики». Актуально для научных, банковских и других организаций.

- **Эмулятор расчета онлайн фичей** позволяет вести расчет, учитывая влияние на фичу происходящих прямо сейчас событий, формируя и агрегируя данные внутри временных окон, которые создаются вокруг искомого объекта.
- **Мониторинг фичей** отвечает за оповещение пользователей и авторов витрин в случае проблем с расчетами (ошибок, высокой нагрузки на кластер и пр.).
- **Поиск фичей** — функционал полнотекстового и точного поиска по всему графу знаний, включающему в себя исходный код, описание, теги, авторов, название и ID
- **Наличие Lineage** — наличие взаимосвязей между моделями и данными. Во время работы с большим количеством экспериментов важно, чтобы в любой момент была доступна информация о том, в каких моделях используются фичи и как логика построения одних фичей зависит от других.

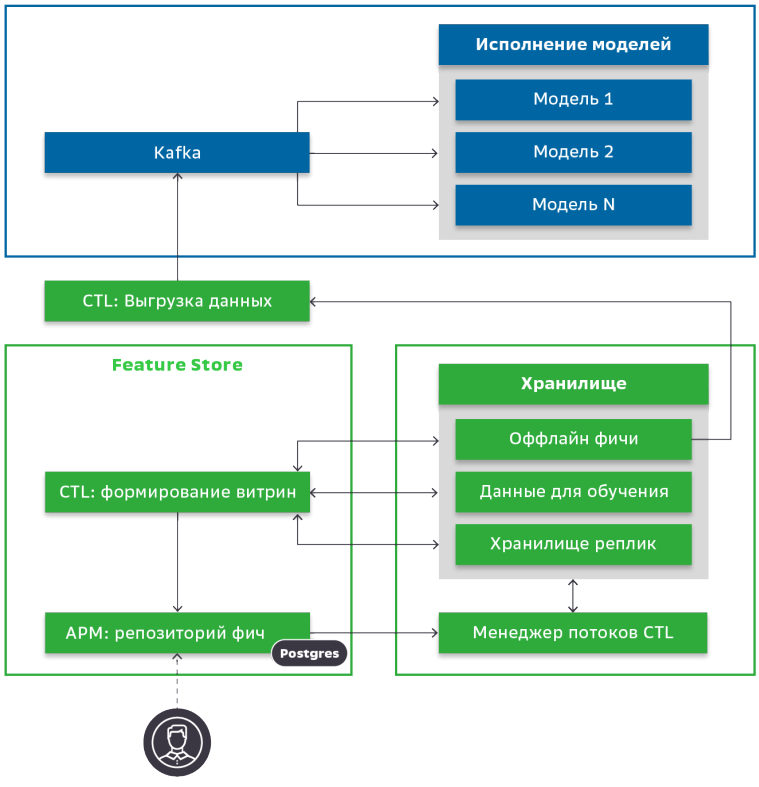


Рисунок 17. *Управление поставкой данных и исполнением моделей*

Архитектура решения включает следующие компоненты (Рисунок 17):

- Хранилище фичей реализуется на основе СУБД PostgreSE — здесь хранится информация о логике построения витрин, связях мехду ними, о пользователях, логи выполнения потоков и все данные, связанные с процессом производства фичей.
- Модуль расчета и экспорта витрин использует технологии CTL (менеджер потоков) и Apache Spark (ядро, в котором происходит расчет витрин).

Использование Feature Store обеспечивает безопасность чувствительных данных, так как для построения логики расчета витрины инженеру не требуется видеть сами данные. Feature Store не отображает содержимое таблиц, расчет производится не внутри API самого инструмента, а делегируется модулю расчета, расположенному на кластере. Обмена данными витрин между кластером и API не происходит.

Управление доступом к функциям Feature Store должно обеспечиваться через ролевые модели, что позволит настраивать видимость объектов между разными группами. Рисунок показывает пример интерфейса сервиса Feature Store (Рисунок 18).

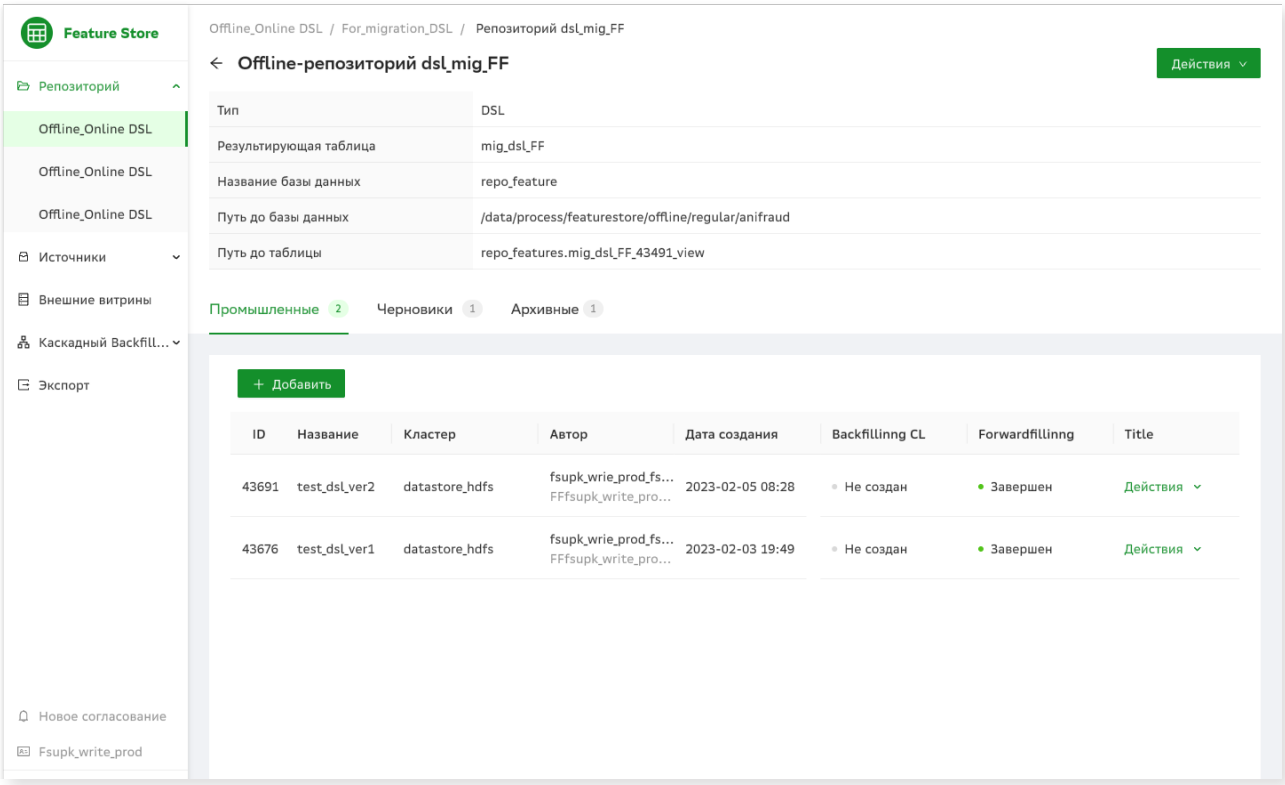


Рисунок 18. *Интерфейс сервиса Feature Store*

4.8 Среда исполнения моделей

Среда исполнения моделей (Model Execution Framework, MEF) — прикладной сервис по сборке, развертыванию и исполнению моделей машинного обучения в форме контейнеров. Сервис дает пользователям возможность разрабатывать AI/ML-модели и обеспечивать их работу в ПРОМе, при этом избежав сложностей, связанных с настройкой инфраструктуры и DevOps конвейера.

Среда исполнения моделей работает с данными в двух ключевых режимах — это stream и database:

- **Stream**: модель получает по HTTP REST запросу (или иному специфицированному протоколу) JSON (или иную структуру) с набором данных. На стороне сервиса модели происходит десериализация полученного запроса, после чего выполняется predict-операция, а результат ее расчета синхронно отправляется в виде HTTP-ответа.
- **Database**: взаимодействие с сервисом модели осуществляется через HTTP-API, однако не передает вектора с данными непосредственно через механизмы HTTP запросов. При этом в коде самого сервиса описан процесс обращения в БД, для получения данных, процесса скоринга, записи результата в БД или выдачи результатов расчета.

MEF обеспечивает единый автоматизированный DevOps-pipeline, в соответствии с текущими стандартами безопасного развертывания образа для всех моделей, разработанных в организации. MEF включает набор задач в Jenkins, которые разворачивают ML-модель, разработанную на языках Java или Python в виде готового контейнера в OpenShift с учетом всех требований ИТ и безопасности к контейнеризованным приложениям. При этом разработчик модели производит минимальную настройку и заказ необходимых компонент для работы модели. Стандартизированный pipeline запускает проверки кода, собирает образ контейнера с моделью и дистрибутив, проставляет флаги прохождения проверок соответствия ИТ и ИБ стандартам, после предоставляет готовый образ для OpenShift — таким образом проделывая все шаги, необходимые для прохождения всех необходимых шагов релизного цикла.

Среда исполнения моделей помимо универсального pipeline

должна обеспечивать следующие возможности:

- набор проверок работоспособности контейнера и модели;
- выбор алгоритма балансировки нагрузки и обеспечения отказоустойчивости;
- встроенный интерфейс для исполнения модели на GPU, а также возможность работы с Multi-instance GPU (MIG);
- возможность настроить вызов модели по определенному расписанию;
- встроенные функции мониторинга и логирования работы модели с возможностью отправки информации во внешние системы;
- возможность из кода моделей обращаться во внешнюю реляционную БД;
- возможность работы с распределенной файловой системой на базе Hadoop;
- предусмотрено batch-исполнение моделей для ускорения времени работы моделей, которым требуются большие объемы данных из БД;
- поддерживается запуск задач для Spark (приложений Spark).

4.9 Сервис управления метаданными

Сервис управление метаданными (СУМД) является одним из ключевых инструментов процесса управления данными (Data governance). Метаданные используются на всех этапах управления и являются основой для современного data-driven подхода «понимай свои данные».

ПРИОРИТЕТЫ УПРАВЛЕНИЯ ДАННЫМИ В АНАЛИТИЧЕСКОЙ ПЛАТФОРМЕ КБ РЕАЛИЗУЮТСЯ ЧЕРЕЗ ОПИСАНИЕ ДАННЫХ, ИСПОЛЬЗУЕМЫХ НА ПЛАТФОРМЕ, ОРГАНИЗАЦИИ К НИМ БЕЗОПАСНОГО ДОСТУПА И ЭФФЕКТИВНОГО ПОИСКА

Организованный процесс работы с метаданными позволяет внести серьезный вклад в эффективность работы организации с данными: сформировать описания данных; повысить качество аналитики; оптимизировать подходы к хранению (ILM); обеспечить повышение качества данных; реализовать возможность переиспользования данных.

Сервис управления метаданными решает следующие задачи:

- каталогизация информации о физических структурах хранения данных;
- формирование описаний данных для удобной навигации и поиска;
- формирование логической модели и обеспечение ее связи с физической моделью данных;

- разметка и тегирование;
- управление глубиной хранения;
- предоставление информации о движении данных (Data lineage) для целей контроля за распространение чувствительной информации;
- визуализация логических и физических структур данных.

Структуры данных, каталогизируемых в СУМД и их бизнес-описания в виде терминов бизнес-глоссария и логической модели данных (ЛМД) используются, в том числе, для обеспечения функции полноценного гетерогенного поиска по платформе при помощи не только наименований физических объектов данных (таблиц/колонок), но и понятных бизнес-пользователям логических наименованиях отдельных сущностей.

ОРГАНИЗОВАННЫЙ ПРОЦЕСС РАБОТЫ С МЕТАДАННЫМИ ПОЗВОЛЯЕТ ВНЕСТИ СЕРЬЕЗНЫЙ ВКЛАД В ЭФФЕКТИВНОСТЬ ИСПОЛЬЗОВАНИЯ ДАННЫХ В ЦЕЛОМ

Для работы с метаданными Платформы пользователям необходим ряд возможностей, которые должна предоставлять система управления метаданными:

- поиск по метаданным в условиях постоянно меняющегося ландшафта и большого количества систем хранения данных;
- работа с физическим и логическим уровнем описания данных;
- хранение истории изменения метаданных;
- учет источников данных, обогащение метаданных информацией о владельце, политике хранения, архивирования, уничтожения;
- маркирование метаданных метапризнаками доступа, уровня критичности, персональных данных и т.д.;
- работа с физической и логической моделью данных для использования в прикладных сервисах и продуктах, обеспечивая их связность по мере движения данных по жизненному циклу.

Основу сервиса составляет концепция «каталога данных» — формирование бизнес-описаний отдельных блоков данных, которые имеются в периметре аналитической платформы. Под блоками данных подразумеваются колонки, таблицы, схемы данных в БД. Следует отметить, что в СУМД не хранятся данные из БД. Сервис собирает только метаданные о структуре хранения, таким образом, исключая риски хранения конфиденциальной информации и обеспечивая под-

ход «минимальной достаточности».

СУМД обеспечивает связь между описаниями (т.н. логическим слоем) и физическими объектами хранения (колонки, таблицы, схемы и т.д.). Сервис в автоматическом режиме на регулярной основе собирает структуры хранения данных с БД. Пользователи имеют возможность через единый UI работать со всеми структурами данных, с применением поиска и фильтрации.

Аналитики составляют бизнес-глоссарий логических атрибутов — описаний данных, сформированных понятным бизнес-языком. Все логические атрибуты связываются (маппируются) на физические объекты – конкретные поля/колонки таблиц данных в БД. Из логических атрибутов формируются логические модели данных (ЛМД), которые позволяет ориентироваться пользователям в информации, которая содержится в платформе.

В сервисе ведутся описания источников данных (карточки источников), загрузка данных которых осуществляется в платформу. Карточки источников связываются с физиче-

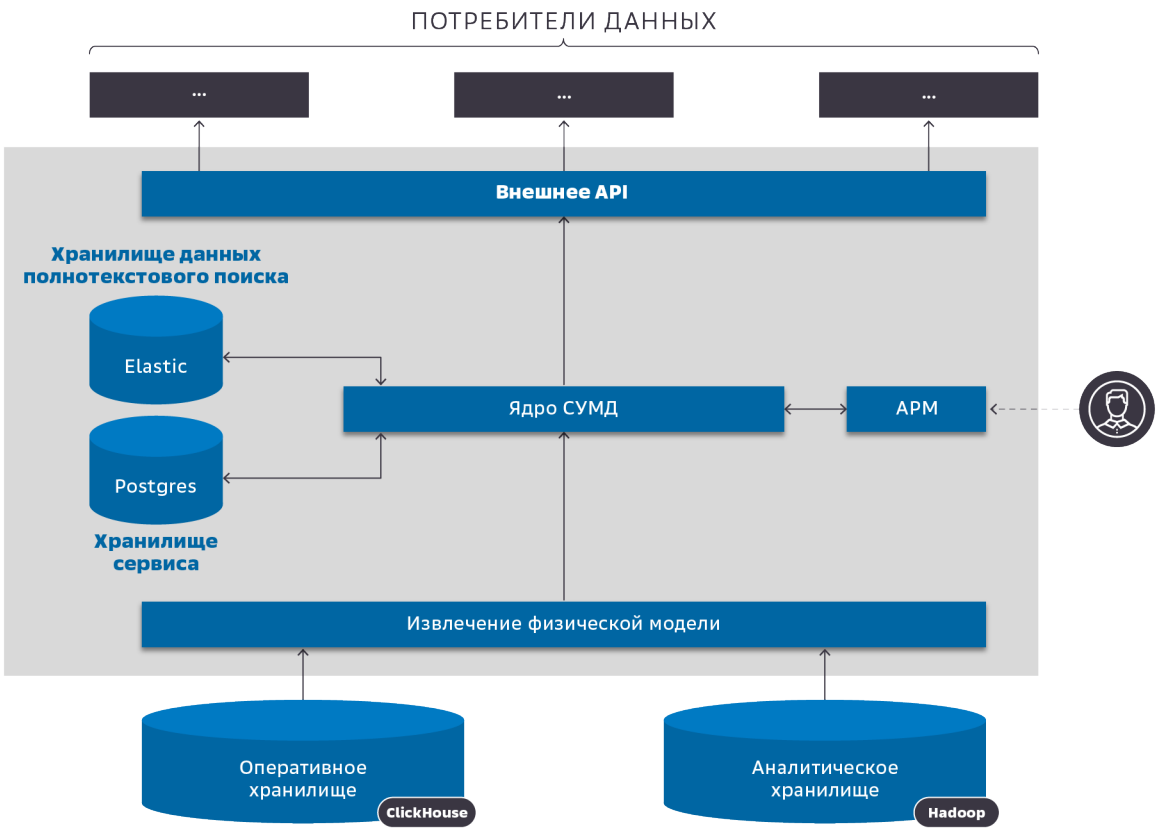


Рисунок 19. *Сервис управление метаданными*

скими объектами в БД, чтобы иметь представление о том, где конкретно находятся данные тех или иных источников (Рисунок 19).

Для управления жизненным циклом данных (ILM) сервис обеспечивает ведение информации о владельцах данных, управление глубиной хранения отдельных блоков данных, сбор статистики об использовании данных на платформе. Отслеживание путей перемещения данных внутри Платформы обеспечивается функциональностью data lineage.

ВСЕ ОПИСАНИЯ ДАННЫХ, СВЯЗИ И КЛАССИФИКАЦИИ ПОСТАВЛЯЮТСЯ В ПЛАТФОРМЕННЫЕ СЕРВИСЫ ПОСРЕДСТВОМ ВНЕШНЕГО API.

Классификация информационных объектов по характеристикам и признакам (необходимым для различных аналитических задач) реализуется с помощью функционала «классификатора». Возможно применение маркирования (тегирования) отдельных информационных объектов необходимыми признаками.

Основой системы является ядро, предназначенное для организации логики взаимодействия между хранилищем метаданных, GUI, инструментами поиска, извлечения и поставки метаданных потребителям. Хранилище метаданных используется для создания и хранения единого реестра объектов проектирования, а движок поиска применяется для обеспечения поиска из пользовательского интерфейса по внутренним объектам данных и должен поддерживать полнотекстовый поиск, нечеткий поиск и фильтрации (Рисунок 20).

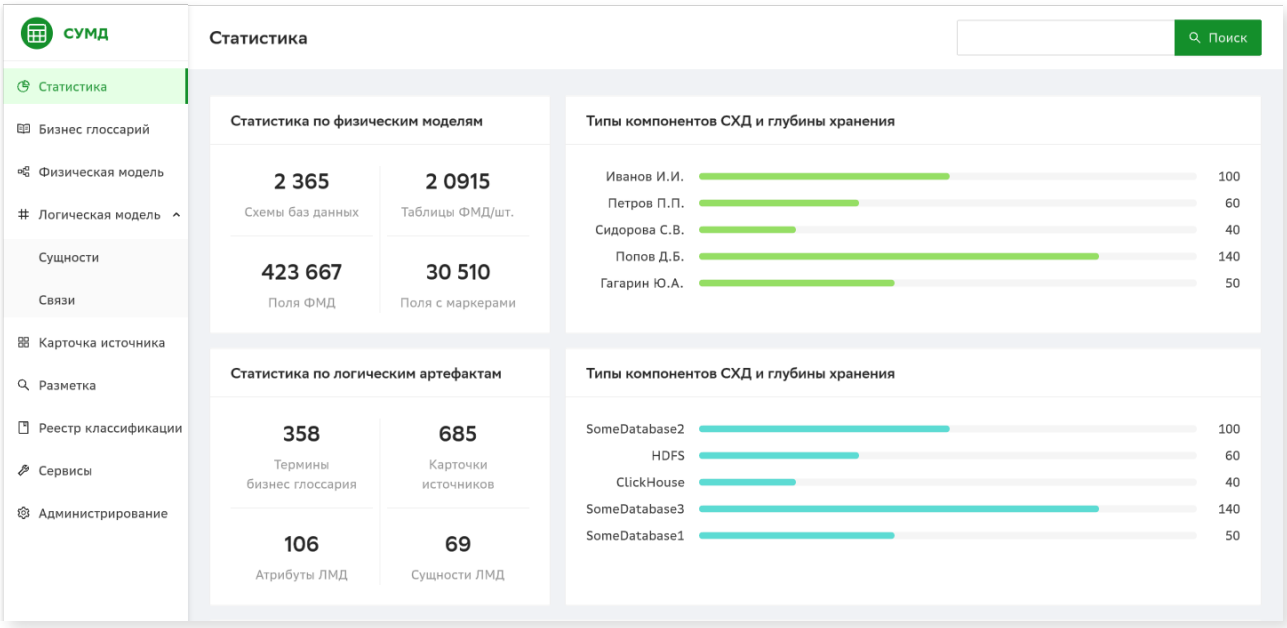


Рисунок 20. *Сервис управление метаданными*

Практика СБЕРа показывает, что информационные объекты, создаваемые в сервисе управления метаданными, востребованы среди сервисов и аналитических команд и позволяют улучшать способность эффективно и безопасно использовать данные, обеспечить прозрачность их использования, сократить время и упростить поиск данных для пользователей разных уровней квалификации. В конечном счете, обеспечение эффективной работы с данными напрямую влияет на поддержание исследований и инноваций в компании, что, в свою очередь, является ключевым фактором для ее конкурентоспособности и долгосрочного роста на рынке.

4.10 Сервис поиска

Важнейшей составляющей аналитической платформы кибербезопасности является возможность оперативно предоставлять данные для анализа в рамках разбора инцидентов и анализа событий аналитиками кибербезопасности. Для этого необходим инструмент, который даст продвинутые поисковые возможности, но при этом не будет требовать от пользователей глубокого знания SQL запросов и понимания внутренней структуры хранилища данных.

Опыт СБЕРа показывает, что большой объем событий не позволяет полноценно использовать существующие средства поиска. Эксплуатация как opensource, так и вендорских решений (Elastic search, OpenSearch, IBM QRadar, Splunk) выявила технологические ограничения по объему обрабатываемых данных и количеству индексированных записей.

Проведя анализ преимуществ и недостатков, было принято решение создать собственный сервис. Имея свой продукт, СБЕР получает возможность делать его «под себя», приоритезировать горящие фичи, используя весь свой накопленный опыт борьбы с киберугрозами, решить проблему импортозамещения и зависимости от производителя.

Основу сервиса составляют:

- конструктор запросов, доступный через UI, который в интерактивном режиме позволяет создавать запросы к хранилищу сотрудникам, не обладающим знаниями SQL

и структуры хранилища;

- ядро поискового движка, который выполняет преобразование поступающего с UI объекта в нативный SQL-скрипт для выполнения в оперативном хранилище.

Поиск в долгосрочном хранилище организован с использованием сервиса доступа к данным. Это позволяет конечным пользователям получать доступ к данным из всех хранилищ в едином интерфейсе (Рисунок 21).

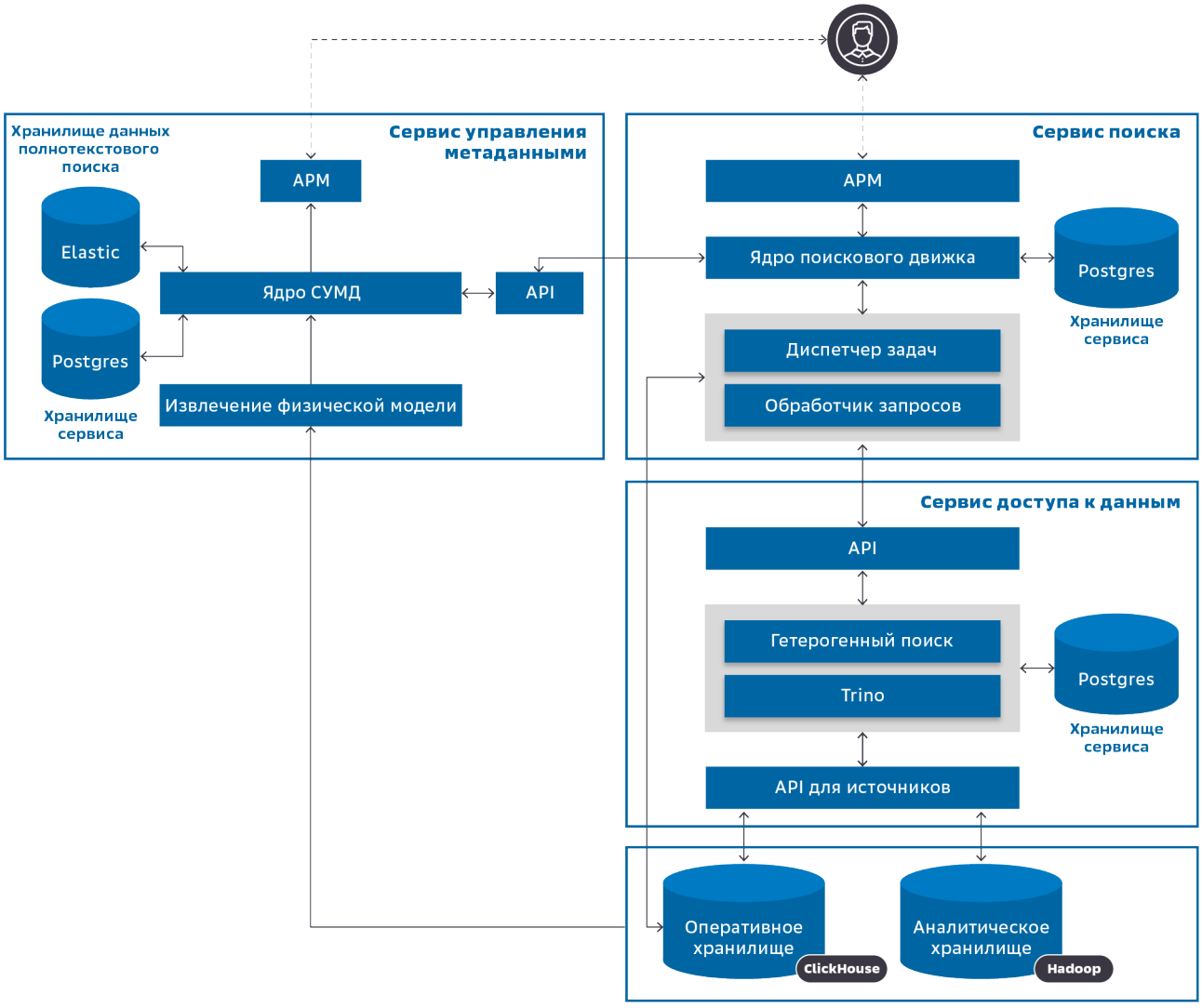


Рисунок 21. *Сервис доступа к данным*

Работа с большим количеством источников предъявляет к пользователям повышенное требование к знанию структуры источника. Интеграция с платформенным сервисом управления метаданными позволяет работать в сервисе использовать как физическую модель данных (ФМД), так и логическую модель данных (ЛМД) при построении поисковых запросов.

По каждой модели данных в рамках сервиса реализованы несколько видов поиска:

- поиск по одному источнику;
- поиск по всему ландшафту хранилища (кросс-поиск).

По запросу от пользователей был реализован механизм шаблонов типовых запросов, так как расследование каждого типа кейсов имеет строго регламентированную последовательность запросов на поиск, в которых меняются только искомые значения. Под каждый типовой запрос создан шаблон доступа, по которому он предоставляется в соответствии с ролевой моделью (Рисунок 22).

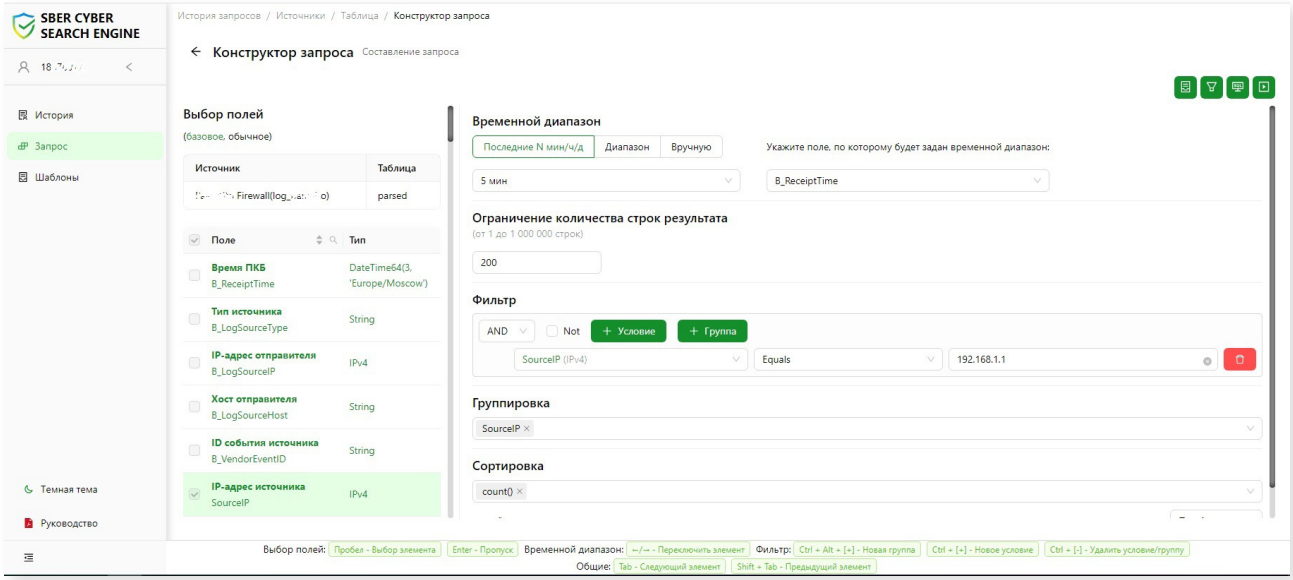


Рисунок 22. *Интерфейс сервиса поиска*

В случае необходимости у пользователя есть возможность в интерактивном режиме использовать SQL-консоль, обеспечивающую валидацию запроса по критериям ИТ и безопасности.

С помощью сервиса поиска были решены следующие задачи:

- гетерогенный доступ к различным хранилищам в интерактивном режиме;
- проведение расследований и пост-анализ инцидентов кибербезопасности;
- возможность проведения сквозного, логического и поиска по вхождению;
- предоставление пользователю информации об источнике;
- режим детального просмотра событий.

Большой объёмом данных в оперативном хранилище СБЕРа (более 5 Петабайт) потребовал детального проектирования архитектуры сервиса поиска и оптимизации хранилища данных.

В результате архитектура сервиса построена на основе асинхронной вычитки поступающих запросов, а в настройках хранилищах был выполнен ряд мероприятий:

- квотирование ресурсов, выделяемых при доступе к хранилищу;
- для «тяжелых» источников использовать кодеки сжатия информации;
- использование индексируемого поля во всех запросах;
- создание «мультикластерной» архитектуры хранилища.

4.11 Лаборатория данных

Лаборатория (Среда исследования и обучения моделей) – единая среда для работы DataScience и аналитиков с доступом к актуальным промышленным данным, инструментами анализа и разработки, а также необходимыми аппаратными ресурсами для работы с использованием технологического стека Big Data.

Лаборатория позволяет: проверять гипотезы, помогает в разработке и обучении AI-моделей, ad-hoc анализе и исследовании данных, вести разработку решений на промышленных необезличенных и немодифицированных данных.

В рамках лаборатории также существует ряд ограничений, которые нацелены на минимизацию рисков влияния на промышленный контур, основные из этих ограничений: лаборатория и ее ресурсы не могут быть участником промышленных процессов, данные или результаты обработки данных в лаборатории не могут поставляться за пределы лаборатории.

Архитектурно лаборатория представляет собой замкнутый контур в промышленном сегменте, в который для проведения исследований копируется необходимый набор данных, пользователи определяют необходимый минимум данных, необходимых для решения производственных задач.

На текущий момент лаборатория для платформы кибербезопасности состоит из 5 типовых экземпляров развертывания (Рисунок 23).

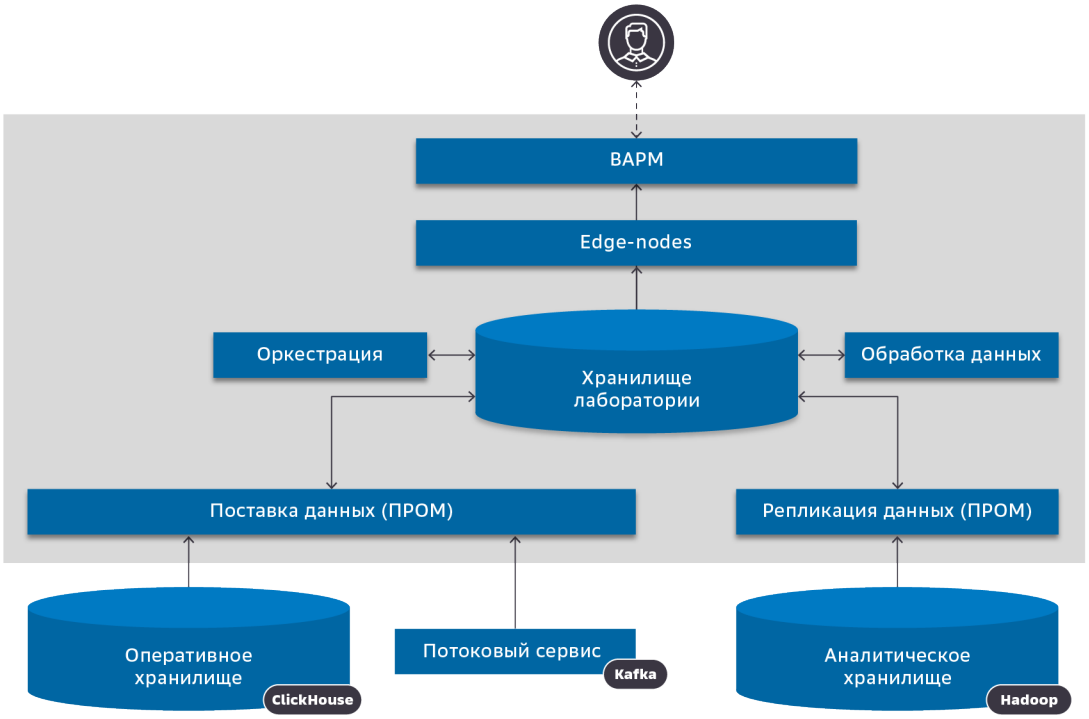


Рисунок 23. *Лаборатория данных*

Для получения ресурсов лаборатории пользователю необходимо сделать несколько простых шагов, определить, какие данные и аппаратные ресурсы требуются для проведения исследования, а также указать, как долго будет проводиться исследование. После анализа потребностей пользователя в автоматизированном режиме для пользователя будет развернут один из 5 типовых экземпляров, после этого лаборатория готова для проведения исследования.

Среда исследования и обучения моделей - это набор сервисов, размещенных в промышленном сегменте, предназначенных для проектирования витрин данных, ad-hoc анализа данных, разработки и обучения ML-моделей на основе промышленных необезличенных данных.

На текущий момент в лаборатории доступны следующие сервисы:

- SberData Platform Hadoop (SDP Hadoop);
- JupyterHub;
- ClickHouse;
- CTL;
- Flink.

SDP Hadoop — это средство обработки и хранения данных, построенное на компонентах экосистемы Hadoop.

В лаборатории настроена федерация для работы с промышленным кластером HDFS и HIVE в режиме доступа, только чтение. Таким образом из лаборатории можно видеть все необходимые Hive-таблицы и HDFS-данные кластера.

Основные инструменты доступные пользователям лаборатории:

- Hadoop (HDFS) — файловая система, предназначенная для хранения файлов больших размеров, поблочно распределённых между узлами вычислительного кластера.
- Hive — система управления базами данных на основе платформы Hadoop с SQL-подобным языком запросов.
- Spark — аналитический движок для обработки больших данных.
- Hadoop User Experience (HUE) — веб-интерфейс для анализа данных.
- Oozie — это серверная система планирования рабочих процессов для управления заданиями Hadoop, используется для CTL-сервиса.

Наличие ClickHouse в Лаборатории может потребоваться в следующих случаях:

- Необходимы данные из промышленного кластера ClickHouse за какой-то определённый период времени для обучения моделей. В таком случае в Лаборатории разворачивается ClickHouse, и с помощью специального CTL-потока копируются данные из необходимой таблицы за требуемый период времени.
- Проведение итоговой проверки потоков Kafka2ClickHouse или каких-либо других взаимодействий с ClickHouse.

Сервис управления загрузкой CTL — это инструмент для управления запуском задач (job) в оркестраторах, а также описание последовательности и условия запуска задач.

Внутри оркестратора задачей (job) может являться:

- приложение YARN (Spark/MapReduce/Tez Job);
- сценарий (sh, cmd, groovy, etc);
- запуск другого приложения;
- запуск другого потока в CTL.

Поток загрузки можно понимать как процедуру с параметрами, которая, в свою очередь, запускает другие процедуры в установленном порядке. Каждый шаг такой процедуры (за-

дача) допускает параметризацию, причем значения параметров задачи могут зависеть от результатов выполнения предыдущих задач.

Задачи, которые решает CTL:

- установка расписания выполнения потоков загрузки;
- определение (установка) событий, факт совершения которых является причиной (триггером) для запуска потока;
- автоматическое вычисление значений параметров, с которыми должен быть запущен поток или задача;
- мониторинг (или отслеживание) активности выполнения (только факт — запущено, завершено, ошибка);
- единое хранилище метаданных процессов загрузки;
- история запуска потоков и их параметров;
- события обновления сущностей;
- статистика обновившихся данных (список обработанных файлов из источника, максимальная дата, по которой проводится выборка дельты изменений, набор бизнес-дат в этой дельте и т.п.).

CTL реализуется как программа, выполняемая в среде JVM. Пользовательские интерфейсы для работы с функционалом представлены: REST API и CTL UI (web-интерфейс). В качестве внутреннего хранилища данных используется база PostgreSQL (Рисунок 24).

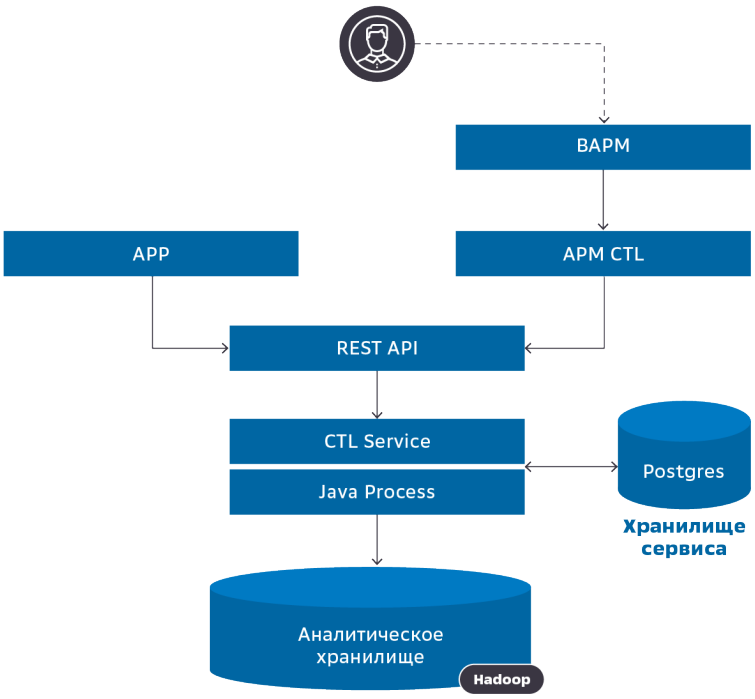


Рисунок 24. *Схема работы CTL*

Apache Flink — это фреймворк и движок для statefull вычислений над неограниченными и ограниченными потоками данных. Flink был разработан для работы во всех распространенных кластерных средах, выполняя вычисления с in-memory скоростью на любом масштабе данных.

Из основных моментов можно подчеркнуть:

- допускается разработка как в параллельном режиме, так и пайплайнами;
- позволяет реализовать как последовательность, так и поток заданий;
- обладает высокой пропускной способностью и низкими задержками;
- поддерживает приложения на Java, Scala, Python и SQL;
- задачи в Flink устойчивы к отказам и используют строго одну семантику.

Flink не обладает собственной системой хранения данных, но использует коннекторы для различных источников данных, таких как Apache Kafka, HDFS, Apache Cassandra, ElasticSearch, Amazon Kinesis и другие.

Для разработки программ есть три основных API: DataStream API, Table API и Python API.

На текущий момент Flink в Лаборатории используется для итоговой отладки разработанных приложений при работе с очередями в Kafka, позволяет проверить: парсинг данных, объединение очередей в одну, перенос преобразованных приложением данных из Kafka в Clickhouse.

4.12 Мониторинг

Развитие аналитической платформы кибербезопасности должно включать в себя комплексный мониторинг метрик и показателей, необходимых для оценки эффективности ее работы, качества данных, бесперебойности работы и анализа потерь.

Для создания системы мониторинга, которая будет охватывать метриками все элементы аналитической платформы, необходимо придерживаться подхода создания многоуровневых контролей, а также скоринга источников, сервисов и продуктов.

Результатами работы такого мониторинга будет кросс-плат-

ПЛАТФОРМЕННЫЙ МОНИТОРИНГ ДОЛЖЕН ПРЕДОСТАВЛЯТЬ АНАЛИТИКУ РАБОТЫ ПЛАТФОРМЫ КИБЕРБЕЗОПАСНОСТИ В РЕЖИМЕ РЕАЛЬНОГО ВРЕМЕНИ С ПРЕДСКАЗАНИЕМ ПОВЕДЕНИЯ КАЖДОГО ЭЛЕМЕНТА НА БАЗЕ РЕТРОСПЕКТИВНОГО АНАЛИЗА, СКОРИНГОВЫХ МОДЕЛЕЙ И КОРРЕЛЯЦИИ МЕТРИК

форменный подход скоринга всех источников, продуктов и сервисов в едином месте с предсказанием поведения.

Для обеспечения комплексного покрытия платформы кибербезопасности необходимыми метриками должны быть реализованы следующие уровни мониторинга (Рисунок 25):

- инфраструктурный мониторинг;
- мониторинг качества данных;
- мониторинг статистического контроля;
- мониторинг синтетического контроля;
- мониторинг потерь;
- скоринговый модуль платформенного мониторинга.

Инфраструктурный мониторинг – неотъемлемая часть любой Платформы по обеспечению базовых контролей работы. В классическом варианте реализации мониторинга инфраструктуры используется Zabbix, в то время как в более современных системах применяются дополнительные системы контроля и визуализации, такие как Prometheus, Grafana, Elasticsearch, Logstash, Filebeat.

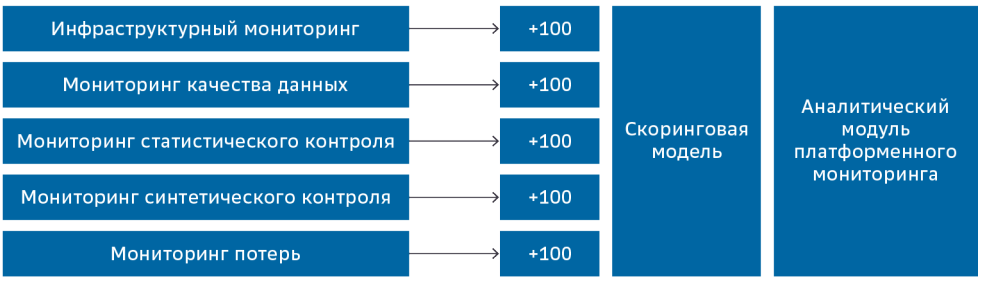


Рисунок 25. *Отображение работы уровней мониторинга*

В зону ответственности инфраструктурного мониторинга входит:

- инфраструктурный мониторинг;
- прикладной мониторинг;
- прикладное логирование (журналирование) ППО/СПО (Log managent);
- визуализация данных мониторинга;
- уведомления о событиях мониторинга;
- автоматическое выполнение процедур (автоинциденты и т.д.);
- трассировка приложений (Tracing);
- предоставление данных инфраструктурного и приклад-

ИНФРАСТРУКТУРНЫЙ МОНИТОРИНГ — БАЗОВЫЙ КОНТРОЛЬ ЛЮБОЙ ПЛАТФОРМЫ, ТРЕБУЮЩИЙ ПОСТОЯННОЙ МОДЕРНИЗАЦИИ ДЛЯ АВТОМАТИЗИРОВАННОГО ОХВАТА НОВЫХ ИСТОЧНИКОВ И МЕТРИК

КАЧЕСТВО ДАННЫХ В ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ ЯВЛЯЕТСЯ КЛЮЧЕВЫМ МЕХАНИЗМОМ, ГАРАНТИРУЮЩИМ ПРАВИЛЬНУЮ РАБОТУ КОРРЕЛЯЦИОННОГО ЯДРА, ЛЮБОЙ ПЛАТФОРМЫ КИБЕРБЕЗОПАСНОСТИ

СТАТИСТИЧЕСКИЙ МОНИТОРИНГ — БАЗОВЫЙ ЭЛЕМЕНТ ПЛАТФОРМЕННОГО МОНИТОРИНГА, КОТОРЫЙ ПРЕДОСТАВЛЯЕТ ПОЛНУЮ КАРТИНУ ОХВАТА ИСТОЧНИКОВ И КОНТРОЛЯ ИХ СОСТОЯНИЯ

ного мониторинга для использования в бизнес-мониторинге и аналитике;
- реактивная и предиктивная Аналитика.

Мониторинг качества данных (Cyber DQ) — система оценки поступающих данных в платформу кибербезопасности и контроль влияния данных на потребителей, в том числе и на SOC, согласно лучшим практикам DAMA.

Мониторинг качества данных можно разделить на следующие уровни:

- **Мониторинг базового качества данных** — обеспечение базового контроля поступающих данных: контроль полей, контроль значений, непрерывность поступления данных;
- **Мониторинг расширенного качества данных** — обеспечение контроля сложных логических зависимостей в данных:
  - корреляция в зависимостях;
  - выявление тренда на снижение поступления данных;
  - ретроспективный анализ поведения потока источнка, в рамках доверительных окон.

Мониторинг статистического контроля (Cyber Base Control) — мониторинг поступления данных от источников. Базовый мониторинг контролирует поступления событий от источника к потребителям в полном объеме, как на уровне событий, так и на уровень потока данных.

Мониторинг синтетического контроля (CyberPing) — механизм генерации и контроля синтетических событий в платформе кибербезопасности, необходимый для повышения уровня доверия к механизмам сбора, обработки, корреляции и хранения. Синтетический мониторинг можно разделить на следующие логические и функциональные уровни:

- уровень генерации синтетических событий источников Платформы;
- уровень генерации синтетических инцидентов информационной безопасности.

Скорринговый модуль платформенного мониторинга — является важнейшим и объединяющим элементом всех уровней мониторинга. На данном уровне происходит:

- профилирование источников;
- итоговая корреляция метрик;

- скорринг источников, Сервисов и Продуктов;
- многоуровневые дашборды, для всех уровней аналитиков, включая операторов SOC, операторов ситуационных центров;
- стратегические панели для руководителей, с онлайн аналитикой работы Платформы, с возможностью погружения до уровня метрики.

Сбор метрик. Одним из основных элементов любой системы мониторинга является слой сбора метрик.

На данном слое происходит:

- сбор метрик с источников;
- сбор статистики и пред-агрегатов с готовых представлений;
- обогащение метрик и событий.

Корреляция и агрегация метрик. Для получения кросс-аналитики по всем источникам, прошедшим уровни мониторинга, необходимо произвести аналитику в едином ядре аналитики с использованием корреляции метрик и событий от источников по единым полям, таким как: HostName, IPaddress и другие.

Визуализация. Для повышения осведомлённости аналитиков ситуационных центров необходимо реализовать многоуровневые дашборды, с возможностью вывода на оперативные и стратегические дашборды в SOC или Fusion Center (Рисунок 26).

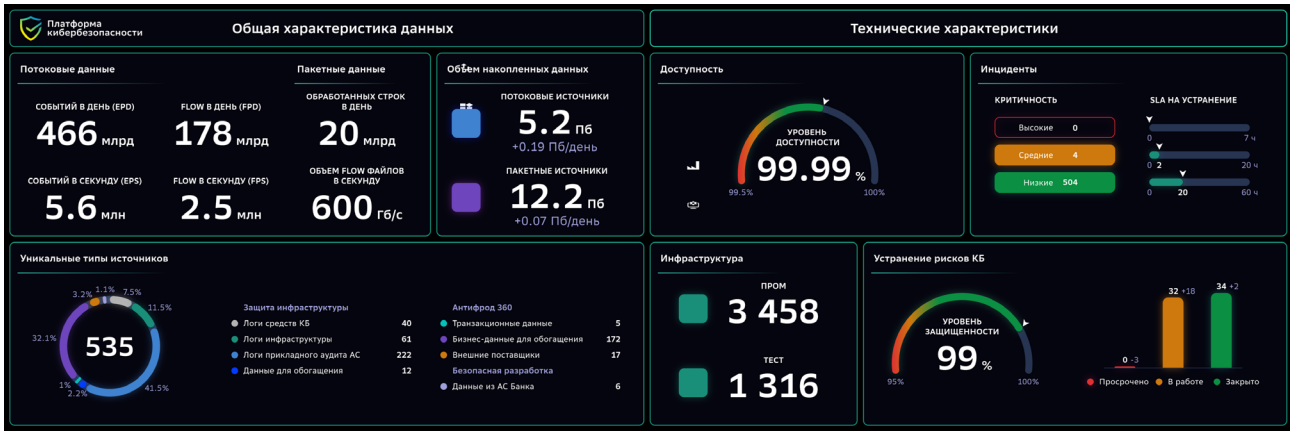


Рисунок 26. *Отображение работы уровней мониторинга*

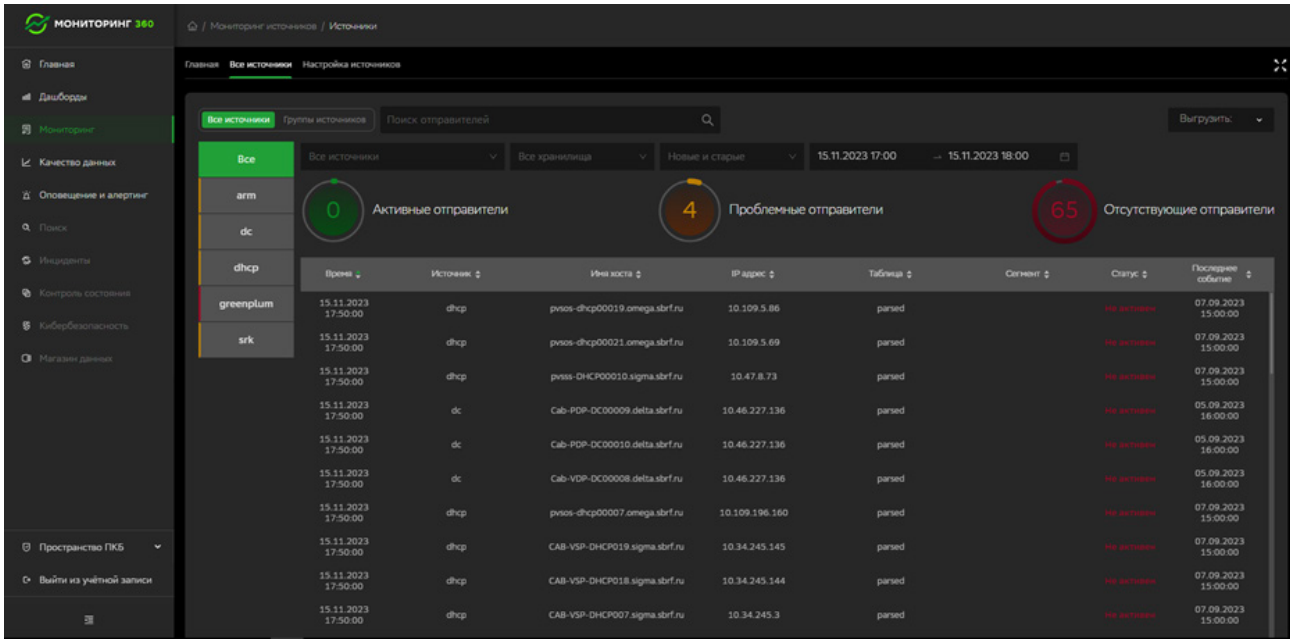


Рисунок 27. *Интерфейс сервиса Мониторинг 360*

Опыт СБЕРа показывает, что современный Платформенный мониторинг обязан покрывать метриками, алертингом и дашбодами все автоматизированные системы и источники, необходимые для полноценной работы SOC в режиме реального времени.

4.13 Использование LowCode в аналитической платформе

Сегодня бизнесу необходимо быть более гибким и адаптируемым, чем когда-либо прежде, чтобы быть конкурентоспособным в быстро развивающейся цифровой среде. Новые модели работы, переход на облачные вычисления и расширение возможностей подключения привели к необходимости изменений в том, как организации создают технологические решения и процессы.

Поскольку бизнес-операции, нетехнические функции, хранение и обмен данными, командная работа и целый ряд других видов деятельности переходят в онлайн, спрос на новые приложения и подключенные услуги резко возрос. Организациям необходимо программное обеспечение для каждой функции, которую можно выполнить на вычислительном устройстве, и в мире просто не хватает инженерного опыта, чтобы удовлетворить этот спрос. Эта нехватка талантливых

специалистов в области программирования и кибербезопасности сейчас ощущается острее, чем раньше из-за скорости цифровой трансформации в последнее десятилетие.

Однако проблема не только в нехватке талантов. Также резко возросла потребность в аналитике больших данных и машинном обучении для повышения эффективности процессов и улучшения бизнес-результатов. Необходимость обрабатывать больше данных на машинной скорости, а также трудности с поиском инженерных знаний для создания новых решений — вот что движет революцией low-code. Считается, что в ближайшее десятилетие развитие low-code будет расти в геометрической прогрессии, а его преобразующий потенциал сравняется с потенциалом облака.

Что такое платформа автоматизации с low-code подходом?

Платформы автоматизации с low-code подходом позволяют пользователям с очень небольшими знаниями в области программирования или техническим опытом создавать или улучшать программные приложения, а также создавать автоматизированные рабочие процессы в визуальных редакторах с возможностью перетаскивания. Платформы с низким кодом поставляются с предварительно созданными модулями, функциями и правилами для распространённых случаев использования и повторяемыми действиями, которые можно быстро комбинировать для создания полноценных сервисов, рабочих процессов и приложений. При необходимости на более позднем этапе они могут быть расширены с помощью настраиваемых функций, написанных вручную более опытными разработчиками.

Автоматизация low-code и автоматизация no-code.

В отличие от автоматизации no-code, разработка с low-code не ограничивает бизнес тем, что позволяет платформа и что уже встроено. Платформа автоматизации с low-code позволяет настраивать код в дополнение к визуальному редактированию и предоставляет пользователям больший контроль над настройкой своих приложений в соответствии с их конкретными требованиями. Это помогает повысить скорость разработки и общую эффективность без ущерба для качества, прозрачности и контроля. У него также больше шансов на принятие более опытными разработчиками, которые могут колебаться в работе над решениями для автоматизации безопасности no-code из-за отсутствия контроля над необработанным кодом и неспособности эффективно отлаживать.

Автоматизация Low-code в сфере кибербезопасности.

Одним из наиболее важных вариантов использования разработки с low-code является автоматизация рабочих процессов, что становится критически важным в кибербезопасности, где скорость реагирования может стать решающим фактором между дальнейшим существованием организации и изнурительной утечкой данных, которая приводит к банкротству. Быстрое обнаружение угроз и реагирование на них имеют решающее значение для обеспечения бесперебойной работы служб и операций, обеспечения безопасности данных, а иногда и спасения жизней, критически важной инфраструктуры и целых городов.

Ни для кого не секрет, что в индустрии безопасности существует огромная нехватка та-

лантливых специалистов, которая усугубляется характером работы и необходимостью постоянной бдительности, выгоранием аналитиков и высокой текучестью кадров. Для эффективной защиты от все более сложных и хорошо организованных субъектов угроз организациям необходима доступность огромных объемов данных об угрозах и возможность фильтровать, анализировать и использовать эти данные со скоростью машины. Аналитики безопасности, даже если они обладают высокой квалификацией, не могут ежедневно обрабатывать все оповещения, генерируемые в современном центре управления безопасностью, без автоматизации и технологической поддержки.

Инструменты и технологии безопасности могут обеспечить положительные результаты в области безопасности только в том случае, если они взаимодействуют друг с другом и взаимодействуют для быстрого обнаружения и устранения угроз. Это означает, например, что инструменту анализа угроз необходимо взаимодействовать с инструментом мониторинга и обнаружения, который, в свою очередь, должен быть подключен к инструменту реагирования, чтобы угроза была быстро и эффективно нейтрализована. Это становится возможным благодаря оркестрации и автоматизации безопасности.

Благодаря автоматизации с low-code команды безопасности могут либо использовать встроенные интеграции, которые есть в платформе, либо легко создавать интеграцию между инструментами кибербезопасности, ИТ и DevOps для оптимизации рабочих процессов. Более тесная интеграция между инструментами также способствует более эффективному сотрудничеству в режиме реального времени и управлению инцидентами.

Внедрение Low-code демократизировало как программирование, так и кибербезопасность, обеспечив большее разнообразие в командах и расширив кадровый резерв, предоставило командам безопасности более индивидуальные возможности автоматизации, чем инструменты автоматизации рабочих процессов без кода. Хотя инженерные навыки будут по-прежнему важны для создания более сложных решений и добавления настраиваемых функций в сервисы и приложения, автоматизация с low-code значительно снижает барьеры для входа в отрасль и повышает общую производительность и удовлетворенность работой как для технических, так и для нетехнических должностей.

Преимущества автоматизации безопасности с низким уровнем кода.

Некоторые из основных преимуществ автоматизации безопасности с низким кодом включают в себя:

Расширение кадрового резерва в области кибербезопасности. Использование автоматизации с низким уровнем кода в операциях по обеспечению безопасности обеспечивает большее разнообразие команд и устраняет некоторые барьеры для входа в систему для будущих специалистов по безопасности. Благодаря автоматизации с низким уровнем кода аналитики любого уровня могут создавать и оптимизировать рабочие процессы, проводить глубокий анализ и расследования, а также добиваться оптимальных результатов в области безопасности.

Снижение усталости от оповещений и выгорания аналитиков. Аналитики безопасности перегружены неуправляемым количеством оповещений, которые приходит-

ся обрабатывать ежедневно. Многие из них являются ложными срабатываниями, и аналитики теряют драгоценное время, работая над данными об угрозах и предупреждениями, которые в конечном итоге оказываются нерелевантными. Автоматизация повторяющихся задач, а также машинный анализ данных об угрозах, корреляция и добавление контекста значительно снижают нагрузку на аналитиков безопасности, которые затем могут работать только с соответствующими оповещениями и угрозами и сосредоточиться на более глубоком расследовании и более продвинутых функциях безопасности.

Снижение затрат на эффективную кибер-защиту. Автоматизация с низким уровнем кода также снижает общую стоимость операций по обеспечению безопасности, устраняя необходимость в сложном программировании или большой команде для создания каждого процесса и функциональности с нуля или для изучения всех предупреждений вручную. Кроме того, благодаря возможности интеграции с другими инструментами в стеке технологий организации эти платформы помогают повысить окупаемость инвестиций от инструментов, которые уже развернуты на предприятии. Автоматизируя большую часть процесса защиты и позволяя даже нетехническим пользователям создавать такие варианты автоматизации, платформы с низким уровнем кода помогают повысить общую производительность и улучшить обнаружение и реагирование без необходимости инвестиций в дополнительные ресурсы.

Оптимизированные процессы и рабочие процессы. Возможность автоматизировать повторяющиеся действия и создавать сценарии для оптимизации процессов делает операции по обеспечению безопасности в целом более эффективными и последовательными. Благодаря процессам расследования и реагирования, изложенным в структурированных схемах, даже неопытные аналитики могут легко понять рабочие процессы и быстро освоиться. Визуальные редакторы сценариев с возможностью перетаскивания упрощают и ускоряют процесс создания сценариев, а также сокращают время окупаемости усилий служб безопасности.

Быстрое обнаружение, расследование и реагирование на угрозы. Возможно, наиболее важным для групп безопасности является то, что автоматизация с низким уровнем кода оказывает явное и прямое влияние на скорость обнаружения, расследования и реагирования на угрозы. Автоматизация позволяет обрабатывать, анализировать и проводить исследования данных на скорости машины с возможностью автоматического запуска ответных действий. Аналитики любого уровня квалификации могут создавать автоматизацию с низким уровнем кода, ускоряя внедрение вариантов использования автоматизации, а также новых функций и возможностей. Общий эффект заключается в значительном сокращении среднего времени обнаружения (MTTD) и среднего времени реагирования (MTTR), а также более быстрых результатов.

Платформа автоматизации процессов с использованием low-code подхода может включать в себя следующие функциональные возможности для поддержки этой функции:

1. Реализацию типовых функциональных возможностей для бизнес-систем данного класса. В случае аналитической платформы кибербезопасности это могут

быть типовые сценарии поставки данных того или иного источника событий, алгоритмы разбора событий, хранения и обработки данных и прочие.
2. Элементы BPMN-системы с возможностью визуального проектирования автоматизируемого бизнес-процесса.
3. Визуальные средства управления конфигурацией типовых функциональных элементов в части настройки и объединения их в единый связный процесс обработки.
4. Визуальные средства конфигурации и управления элементами обогащения данных, в том числе путем использования типовых интеграционных сценариев с внешними системами, а также элементов логики принятия решений и управления правилами.
5. Средства работы с кодом, в общем случае на специализированном DSL (domain specific language) для обеспечения гибкости при реализации ряда функциональных возможностей там, где алгоритм может или должен быть определен пользователем системы, а возможностей визуального управления и конфигурирования недостаточно.

DSL — особый построенный язык для решения задач специализированного бизнес-домена. Например, вы разрабатываете ПО для обработки событий кибербезопасности. В этом случае бизнес-домен может включать такие понятия, как сетевой шлюз или список контроля доступа (ACL). Вы можете создать свой DSL язык для решения задач этого домена. DSL можно рассматривать как программный интерфейс для взаимодействия с объектами вашего бизнес-домена.

DSL не является языком программирования общего назначения. Он ограничивает набор действий и состав доступных ключевых слов в заданной области, что позволяет частично отказаться от использования Java или Python, уделить больше внимания решению самой бизнес-задачи. Низкоуровневые детали реализации DSL должны быть скрыты, что делает DSL более привлекательным для использования непрофильными для ИТ специалистами.

Среди различных DSL можно выделить следующие категории:

1. **Встроенный DSL**. Для примера, если ваша платформа использует Java как основной ЯП, то встроенный DSL будет реализован на Java. Исторически сложилось, что Java не рассматривался долгое время как DSL-дружественный ЯП, так как её громоздкий и не гибкий синтаксис усложняет построение интуитивно понятного, выразительного DSL. Это ограничение было по большей мере снято с появлением в Java8 лямбда-выражений.
2. **Комбинированный DSL**. Этот тип DSL более характерен для платформ, использующих jvm-ориентированные ЯП. В настоящий момент существует около 100 ЯП, для которых целевой платформой выполнения является java virtual machine. Многие из этих ЯП моложе, чем сама Java, поэтому были спроектированы с меньшим количеством ограничений и более лаконичным синтаксисом.
3. **Внешний DSL**. В этом случае вы разрабатываете свой собственный язык со своим уникальным синтаксисом и семантикой. Для этого необходимо также подготовить отдельный инструментарий для синтаксического разбора и компиляции кода, написанного на этом DSL. Это требует

не только больших вложений ресурсов, но и является непростой задачей. Преимуществом данного подходя является максимально неограниченные возможности, которые можно реализовать при проектировании данного DSL.

Рассмотрим варианты реализации low-code платформы.

1. **Интерпретация**. Определение интерпретатора по Таненбауму звучит так: интерпретатор — это программа на языке Я0, которая берет программы, написанные на языке Я1 в качестве входных данных, рассматривает каждую команду по очереди и сразу выполняет эквивалентный набор команд языка Я0. Таким образом, задача разработчика — определить, чем являются языки Я0 и Я1, разработать алгоритм выполнения инструкций языка Я1. В случае, если необходимо реализовать автоматизированное выполнение какого-либо бизнес-процесса, то Я1 обычно является одним их популярных языков разметки, например JSON, Yaml или XML. Используя один из этих инструментов, удобно хранить цифровое представление процесса, разделяя его на шаги и управляя поведением этих шагов, задавая значения предопределенным параметрам. Если же речь идет об использовании DSL для автоматизации процесса, то наиболее распространенной ситуацией применения метода интерпретации является разработка внешнего DSL, который и будет выступать языком Я1, а интерпретация используется для удешевления реализации такого DSL.
2. **Трансляция** — это замена каждой команды программы на языке Я0 эквивалентному набору команд на языке Я1. В случае, если мы автоматизируем бизнес-процесс, состоящий из набора типовых шагов, и при этом есть жесткие требования на время ответа системы или ее пропускную способность, наиболее правильным способом реализации будет использование метода компиляции. Java предоставляет богатый выбор вариантов для компиляции исходного кода и подгрузки Java классов в процессе выполнения программы.
3. **Гибридный метод**. Интерпретация с внедрением в ход исполнения скомпилированных фрагментов.

4.12 Обеспечение защиты платформы

Обеспечение кибербезопасности платформ больших данных стало неотъемлемой частью сферы информационных технологий. В современном мире объемы данных растут экспоненциально, управление и обеспечение безопасности этих данных становится задачей критического значения. Важной характеристикой платформ больших данных является их способность масштабирования, что позволяет организациям обрабатывать данные в реальном времени и извлекать ценные инсайты. Однако вместе с преимуществами, связанными с обработкой больших данных, платформы также представляют уникальные вызовы в области кибербезопасности. Обеспечение кибербезопасности таких систем подразумевает многоуровневый подход, возникают комплексные задачи как по защите самих данных, так и пользователей, работающих с этими данными. При этом применяемые решения для обеспечения безопасности не должны сказываться на производительности систем и, тем более, делать невозможной

аналитическую обработку, ведь именно это и является целью создания озера данных.

Ввиду того, что большие объемы данных часто содержат чувствительную информацию, такую как личные данные клиентов, конфиденциальные и корпоративные данные и другие ценные ресурсы, они становятся целью для злоумышленников. Угрозы могут включать в себя атаки на безопасность данных, сбои в работе систем, фишинговые атаки, вредоносное программное обеспечение и другие виды кибератак. Эти атаки могут привести к серьезным последствиям, включая утечку информации, финансовые потери, несоблюдение законодательства и требований регуляторов в области защиты данных, репутационному ущербу организации.

С точки зрения классических подходов организация защиты Платформы сводится к выполнению требований безопасности как на системном, так и на прикладном уровнях. На системном уровне необходимо обеспечить безопасность сетевой инфраструктуры, серверов, операционных систем, СУБД и хранилищ данных, серверов приложений. К основным механизмам безопасности прикладного уровня относятся: администрирование и управление доступом, идентификация и аутентификация, контроль целостности, конфиденциальности и доступности, криптографическая защита, аудит.

В отличие от коробочных решений, изначально включающих инструменты защиты иразграничениядоступа,Платформастроится на основе OpenSource компонентов, для которых отсутствует единое решение, полностью удовлетворяющее нашим потребностям. Это привело к необходимости разработки гибридных решений, состоящих из общедоступных компонентов и элементов защиты собственной разработки. Все эти факторы в совокупности накладывают повышенные требования к специалистам, разрабатывающим и сопровождающим Платформу.

Использование OpenSource компонентов, стандартные настройки которых часто не обеспечивают достаточной защиты, ошибки в конфигурациях открытого ПО, приводят к появлению брешей в системе.

В этом разделе мы рассмотрим ряд лучших практик, которые помогут организациям обеспечить безопасность своих платформ больших данных. Эти практики включают в себя методы и стратегии, которые можно реализовать для повышения уровня кибербезопасности и уменьшения рисков.

Разработка и реализация политики кибербезопасности является первым и важным шагом к защите платформ больших данных. Политика кибербезопасности должна устанавливать правила и стандарты для доступа к данным, аутентификации, шифрования, мониторинга, аудита и других аспектов безопасности. Она также должна определять ответственность сотрудников и процедуры реагирования на инциденты.

Внутренние и внешние угрозы для аналитической платформы кибербезопасности

На платформах больших данных необходимо учитывать разнообразные угрозы, как внутренние, так и внешние. Эти угрозы могут иметь различные характеристики и мотивации, но их потенциальные последствия для безопасности данных могут быть значительными.

Одной из значительных угроз кибербезопасности на платформах больших данных являются угрозы, исходящие изнутри организации (внутренние угрозы). Это могут быть злонаме-

ренные действия сотрудников или их неосторожность, которые могут привести к утечке данных. Важно регулярно контролировать доступ сотрудников к данным, а также проводить обучение правилам безопасности.

В ситуации, когда с системой работают высококвалифицированные специалисты, при разработке модели нарушителя и угроз требуется уделить повышенное внимание дополнительным средствам контроля за возможным внутренним нарушителем. С учетом этих факторов формулируются модели нарушителя и угроз платформы, проводится оценка рисков и выбираются меры по устранению или смягчению угроз.

В связи с протекающими в мире событиями активизировались экстремистские и преступные группировки, атаки совершают не только отдельные физ. лица, действующие по идеологическим или политическим мотивам, но и полноценные кибервойска иностранных государств, действующих для дестабилизации работы гос.органов и организаций. Они способны внедрить дополнительные функциональные возможности в коммерческое и OpenSource ПО. Эти группы относятся к внешним нарушителям. Защита от внешних угроз требует развертывания средств защиты и мониторинга для раннего обнаружения аномальной активности (Рисунок 28).

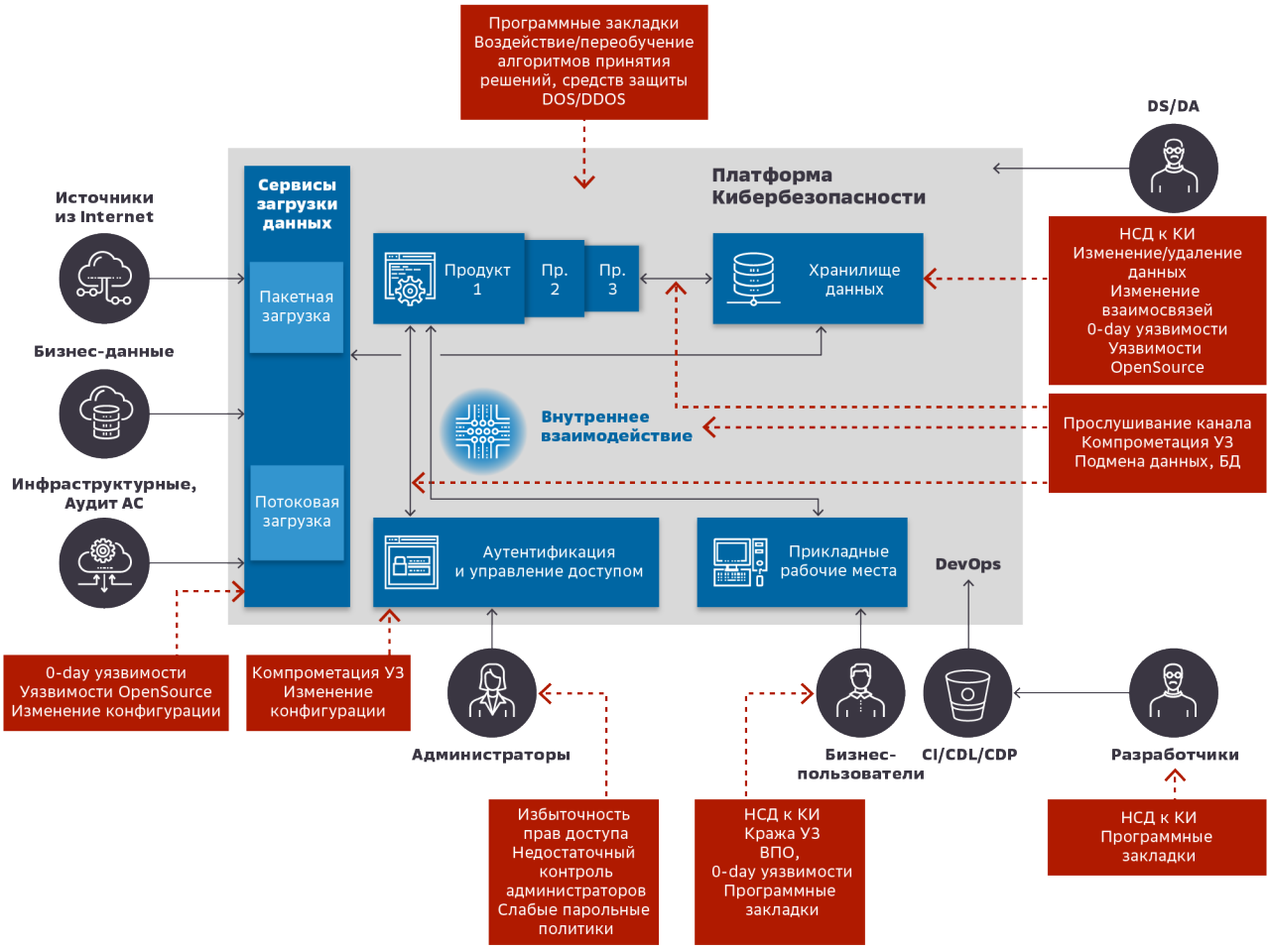


Рисунок 28. *Основные угрозы безопасности*

Безопасность платформы складывается из следующих компонентов (Рисунок 29)

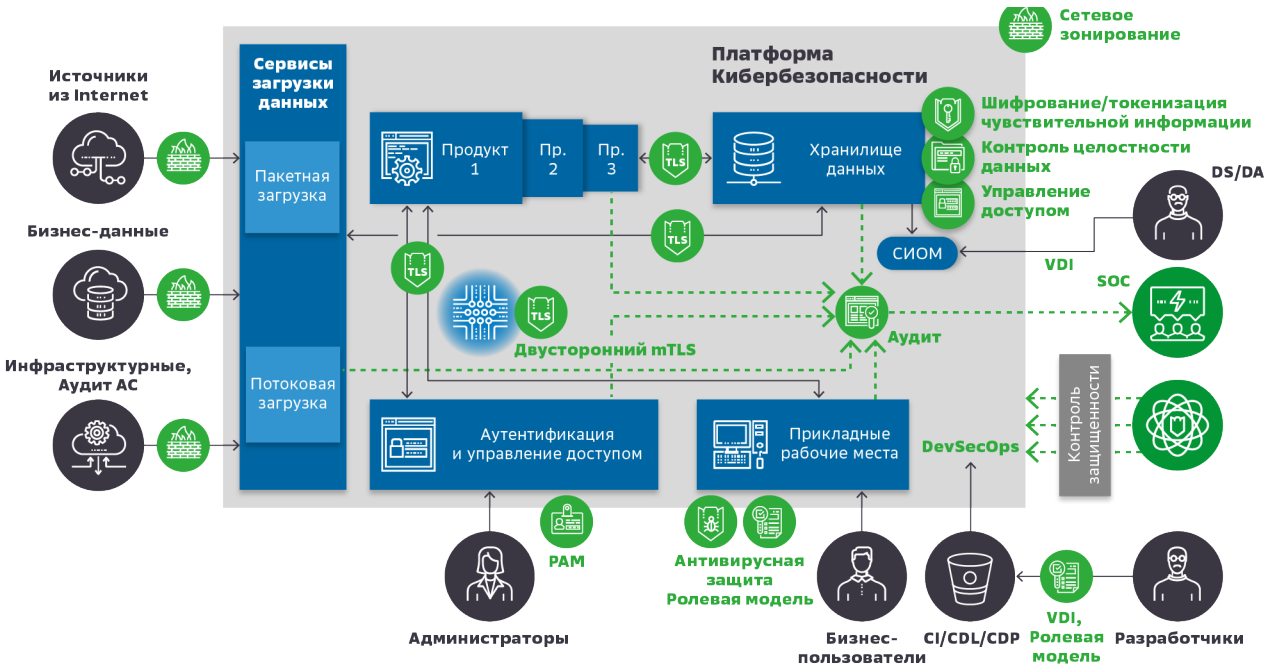


Рисунок 29. *Элементы защиты платформы*

DevSecOps и методологии разработки безопасных приложений

Процесс разработки, отладки и тестирования работы нового ПО ведется с применением инструментов и методологий безопасной разработки DevSecOps в единой среде разработки с использованием синтезированных или обезличенных данных. Подходы DevSecOps обеспечивают минимизацию рисков безопасности на всех стадиях разработки приложений.

В процесс безопасной разработки Application Security (AppSec) включены как ручная экспертиза (анализ архитектуры, code review, пентесты), так и автоматизированные проверки (SAST, OSS, DAST).

Интеграция кибербезопасности в разработку и эксплуатацию платформ больших данных обеспечивает более эффективное управление рисками и уменьшает вероятность возникновения угроз.

Защита сетевого контура платформы

Классическая защита контура и инфраструктуры обеспечивается использованием межсетевых экранов, WAF, отделяющих Платформу от пользовательского сегмента, сред разработки, тестирования и других автоматизированных систем банка.

Задача обеспечения защиты контура осложняется тем, что компоненты Платформы распределены между несколькими ЦОДами, при этом используется как физическая, так и виртуальная инфраструктура. Для сетевого разграничения применяются как аппаратные межсетевые экраны, так и инструменты программно-определяемых сетей. Элементы Платформы разделены на логические группы, доступ к которым ограничен минимально необходимым набором сетевых взаимодействий.

Защита ОС и борьба с уязвимостями

Обеспечение регулярного обновления ПО для всех компонентов платформы (Patch management), безопасное конфигурирование ОС, прикладного ПО является обязательным аспектом кибербезопасности. Это довольно трудоемкий процесс, когда кластеры хранения и обработки включают сотни машин с непрерывно протекающими бизнес-процессами по защите банка и средств клиентов. На постоянной основе проводятся сканирования инфраструктуры при помощи сканеров уязвимостей. Выполняется работа по выявлению и анализу новых угроз кибербезопасности (TIP — Threat Intelligence Platform), анализу рисков, проведению проверок на применимость угроз.

Организация доступа пользователей

Учитывая большое разнообразие пользователей и обрабатываемых данных, особое внимание необходимо уделять вопросу организации доступа пользователей к сервисам Платформы.

Разграничение полномочий обеспечивается применением ролевой модели (RBAC). Пользователям присваиваются роли, а затем предоставляются соответствующие полномочия по доступу к данным. Это позволяет разделить пользователей на категории в соответствии с выполняемыми ролями, уйти от назначения детализированных полномочий каждому пользователю, соответственно упростить управление разрешениями и снизить нагрузку на администраторов. Кроме того, это сокращает вероятность возникновения ошибочного и непреднамеренного предоставления доступа.

Ролевая модель постоянно адаптируется под потребности продуктов и сервисов с использованием гранулярного определения полномочий. При предоставлении доступа к данным учитывается информация из сервиса управления метаданными, который предоставляет инструменты тегирования данных признаками доступа, уровня критичности, принадлежности к ПДн и т.д.

При организации доступа пользователей приходится решать множество проблем. Распространенной проблемой является неполное понимание бизнес-задач подразделений и перечня данных, к которым им действительно необходим доступ. При назначении ролей должен применяться принцип минимальных полномочий. Но со временем полномочия и доступы пользователей могут расширяться. Это приводит к предоставлению пользователям (администраторам, разработчикам, data-инженерам, data-аналитикам, data-сайентистам, бизнес-пользователям и т.д.) излишних прав и включению их в более привилегированные группы, чем это необходимо для выполнения бизнес-функций, что дает повышенные права злоумышленникам в случае компрометации учетных записей. Для решения этого вопроса необходимо проводить периодические аудиты предоставлен-

ных прав доступа и вносить корректировки.

Защита данных при хранении и передаче (криптографическая защита)

Помимо решения вопроса организации доступа, важно обеспечить и надежное хранение критичной информации. Для обеспечения защиты данных при хранении применяется шифрование.

Также на Платформе применяются методы маскирования, токенизации и обезличивания данных в процессах обработки, хранения и экспорта для конфиденциальных и персональных данных, данных банковских карт.

Для обеспечения защиты данных при передаче между сервисами Платформы, а также при взаимодействиях с другими системами, используются соединения, защищенные HTTPS/TLS с двусторонней аутентификацией по сертификатам, выпущенным удостоверяющим центром банка.

Аудит и мониторинг событий кибербезопасности

Регулярный аудит кибербезопасности платформы помогает выявлять потенциальные слабые места и уязвимости. Этот процесс включает в себя сканирование на наличие уязвимостей, анализ журналов событий и проверку соблюдения политик безопасности. Результаты аудита позволяют разработать и реализовать необходимые меры безопасности.

Для решения комплексной задачи, связанной с вопросами аудита и мониторинга событий кибербезопасности, необходимо реализовать собственный модуль аудита, обеспечивающий протоколирование событий безопасности на уровне инфраструктуры, ОС, СУБД, Платформы (технологические сервисы, операции доступа пользователей к данным), а также на уровне бизнес-приложений (действия пользователей в UI прикладных продуктов) с настраиваемым уровнем детализации. Модуль аудита должен обеспечивать возможность проведения расследований и соответствовать всем требованиям кибербезопасности по разграничению доступов, глубине хранения.

Все зарегистрированные события безопасности передаются в сервис аудита и в единый SOC Сбера для последующей обработки и реагирования на возможные инциденты в соответствии с плейбуками.

Для отслеживания и своевременного реагирования на IT-инциденты компоненты Платформы подключены к системе IT-мониторинга, что минимизирует сроки решения возникающих проблем. Для обеспечения целостности потоков данных, в реальном времени проходящих через Платформу, настроены уведомления по набору метрик этих потоков.

5. Уровень зрелости платформы

5.1 Метрика оценки аналитических платформ кибербезопасности

Модель зрелости — это совокупность преимуществ, критериев и уровней развития платформы по ключевым для всей организации направлениям и практикам, на развитии которых необходимо сосредоточиться.

Переход между уровнями обеспечивается за счёт выполнения критериев по направлениям разработки, тестирования, внедрения и надёжности. Например: полнота загруженных данных, качество данных, наличие и зрелость технологических инструментов, уровень надежности компонент.

Модель зрелости обладает дифференцированным подходом к работе с требованиями платформы и ее компонент. Критерии показывают фактический уровень зрелости платформы и помогают лидерам определить, на каких технологиях и практиках производственного процесса необходимо сосредоточиться.

Для оценки используется, как правило, количественный показатель, выраженный в процентном отношении текущего количества выполненных мер к максимальному значению количества реализованных.

В качестве метрик для оценки платформы кибербезопасности можно выделить следующие глобальные группы метрик:

1. **ДАННЫЕ**
   - **Покрытие источников**
     полнота загруженных данных, т.е. платформа обладает всеми необходимыми данными для работы потребителей.
   - **Мониторинг метрик по данным**
     Весь процесс загрузки и обработки данных покрыт необходимыми метриками мониторинга (например, анализ потерь, контроль поступления событий, качество

данных, анализ отклонений, мониторинг аномалий, мониторинг работы всего процесса загрузки и обработки «business health»).

- **Логическая и физическая модель**
  Полнота описания данных, т.е. все данные платформы описаны и находятся в актуальном состоянии для потребителей.
- **Data Governance**
  Уровень зрелости процессов управления данными (например, объем реализованных практик DAMA DMBOK).

2. **ТЕХНОЛОГИЧЕСКИЕ СЕРВИСЫ**

   Уровень зрелости технологических сервисов, т.е. готовность технологического сервиса к выполнению поступающих задач на постоянной основе,
   low code / no-code, инструменты администрирования сервиса. Ниже перечислены обязательные сервисы, которые должны входить в состав платформы:
   - Загрузка;
   - Хранение и обработка;
   - Выгрузка;
   - Платформенный доступ к данным;
   - Поиск;
   - Управление метаданными;
   - Управление доступом;
   - Комплексный мониторинг.

3. **ИНСТРУМЕНТЫ AI**
   - **Управление поставкой данных и исполнение моделей.** Инструменты исполнения AI-моделей в ПРОМ (например, готовый базовый фреймворк для исполнения моделей, реестр фичей и моделей, DevOps конвейер, унификация процесса разработки моделей и вывода в ПРОМ).
   - **Среда исследований и обучения моделей.** Инструменты для создания, обучения, запуска AI-моделей для Data Science, Data Engineer и аналитиков, инструменты поставки данных.

4. **НАДЕЖНОСТЬ**

   способность системы предоставлять согласованные

и качественные услуги, выполнять задачи и обрабатывать запросы пользователей в рамках согласованных параметров производительности.

- **Интегральный уровень защиты компонентов**
  показатель, который отражает общую степень защиты компонентов системы от различных угроз и уязвимостей. Определяется на основе анализа множества факторов, таких как описание функционала, архитектура, ВНД и регламенты, управление ИТ-активами и конфигурациями ИТ-активов, управление уязвимостями, управление инцидентами, управление доступом (аутентификация и авторизация), управление разработкой и ЖЦ объекта, защита сети, защита каналов взаимодействия, защита от DDoS, антивирусная защита, защита данных, контроль привилегированного доступа, криптографическая защита, мониторинг и аудит (управление инцидентами), контроль и анализ защищенности.
- **Доступность**
  свойство системы или сервиса быть доступным и готовым к использованию в необходимый момент времени и в нужном месте, с требуемым качеством и уровнем производительности.

Визуализация состояния платформы кибербезопасности позволяет аналитикам графически представлять текущее состояние зрелости платформы и оценивать потенциальный эффект от реализуемых мер. При этом одна из важных задач при визуализации метрик — это выбор наиболее наглядной графической модели для конкретного случая (Рисунок 30).

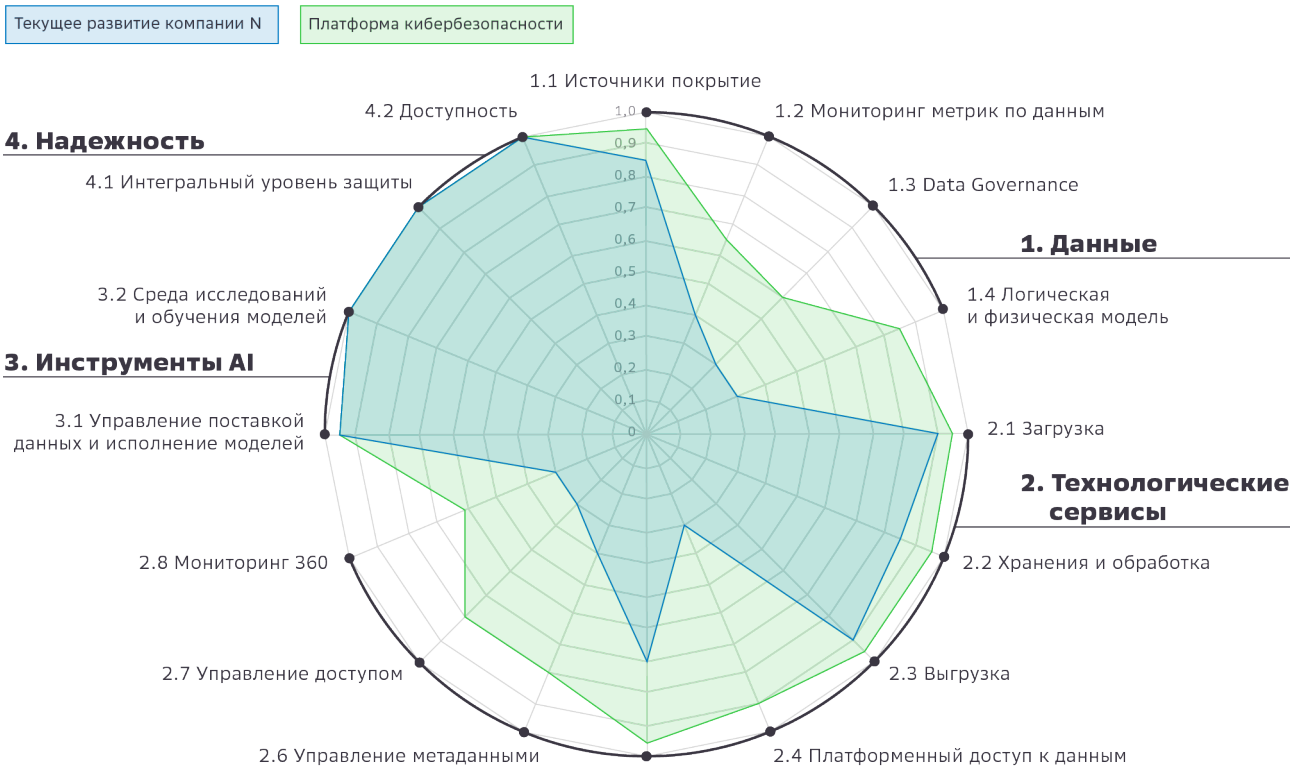


Рисунок 30. *Уровень зрелости платформы*

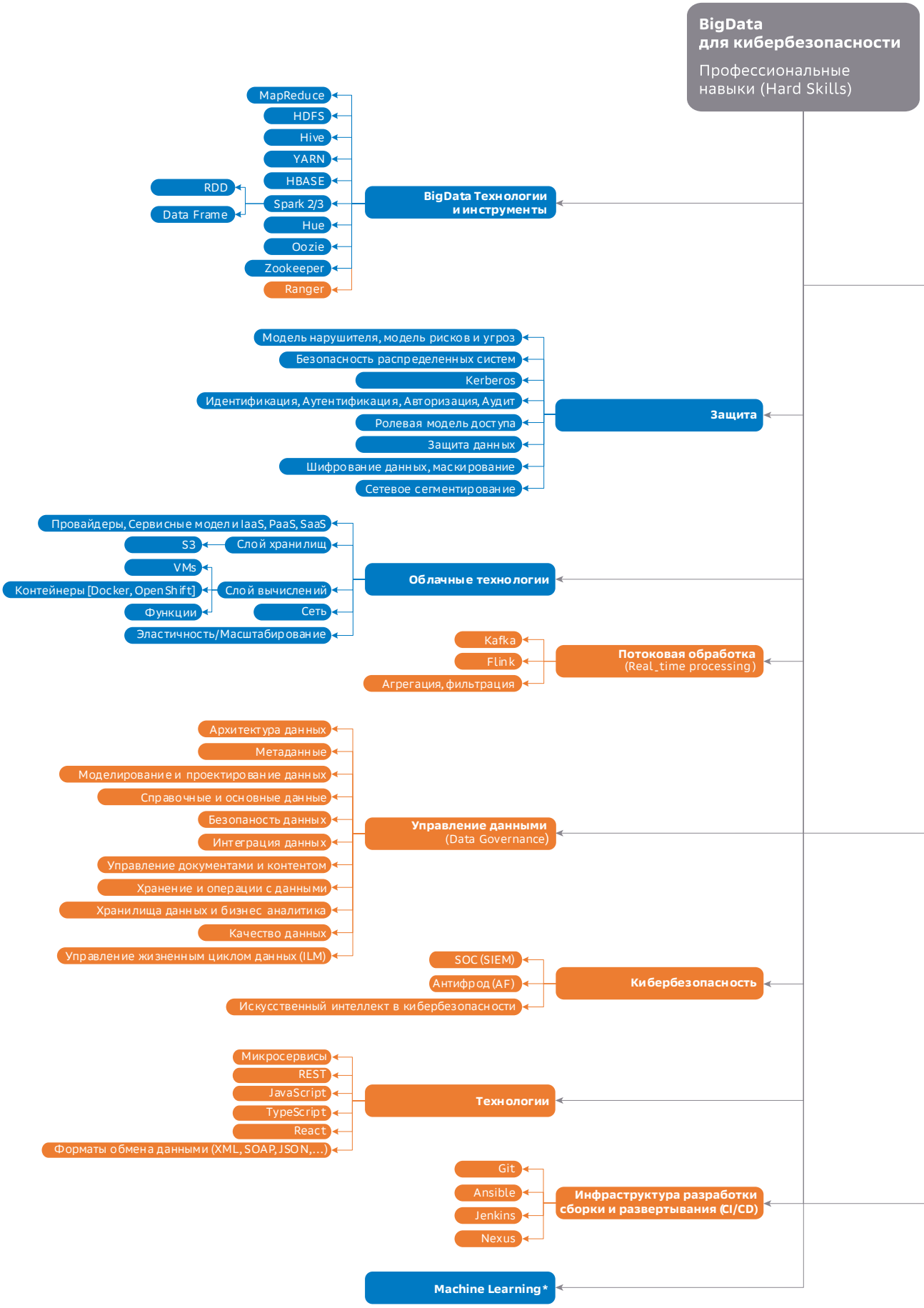
5.2 Дерево компетенций для разработки и эксплуатации

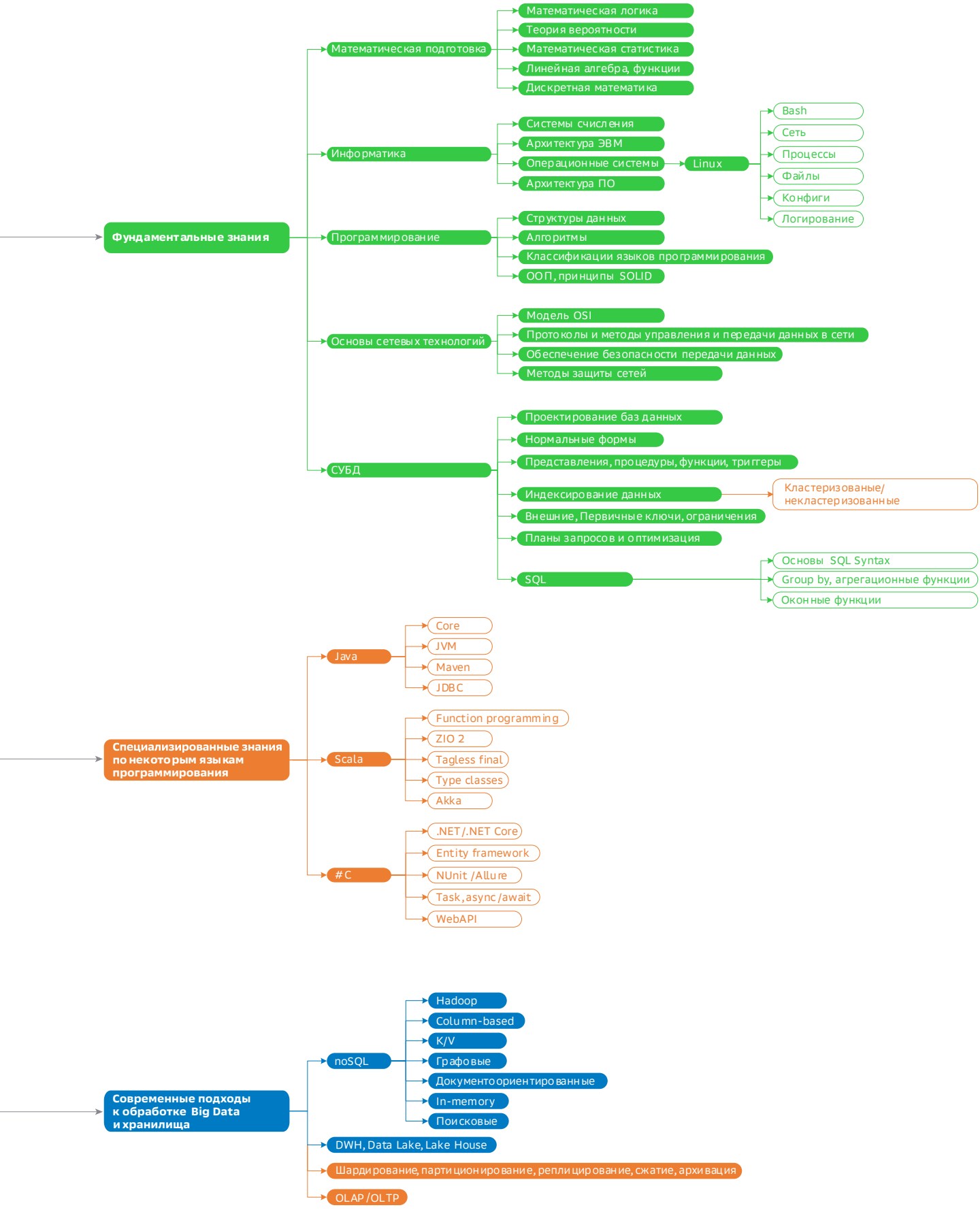
Дерево компетенций для разработки и эксплуатации платформенных решений описывает необходимый набор знаний, навыков и опыта, требующийся для разработки и эксплуатации. Включает в себя различные уровни компетенций, начиная от базовых навыков программирования и заканчивая сложными навыками управления проектами и обеспечения безопасности.

Дерево компетенций состоит из нескольких уровней, начиная от базовых навыков и заканчивая более сложными и специализированными навыками. Оно может быть использовано для оценки квалификации специалистов, определения их уровня знаний и навыков, для развития персонала и планирования обучения. Кроме того, оно может помочь компаниям в выборе наиболее подходящих кандидатов.

Дерево компетенций, представленное ниже, построено на основе опыта СБЕРа при разработке аналитической платформы кибербезопасности (Рисунок 31).

Рисунок 31. *Дерево компетенций*




Фундаментальные знания
Математическая подготовка
Математическая логика
Теория вероятности
Математическая статистика
Линейная алгебра, функции
Дискретная математика
Информатика
Системы счисления
Архитектура ЭВМ
Операционные системы
Linux
Bash
Сеть
Процессы
Файлы
Конфиги
Логирование
Архитектура ПО
Программирование
Структуры данных
Алгоритмы
Классификации языков программирования
ООП, принципы SOLID
Основы сетевых технологий
Модель OSI
Протоколы и методы управления и передачи данных в сети
Обеспечение безопасности передачи данных
Методы защиты сетей
СУБД
Проектирование баз данных
Нормальные формы
Представления, процедуры, функции, триггеры
Индексирование данных
Кластеризованые/ некластеризованные
Внешние, Первичные ключи, ограничения
Планы запросов и оптимизация
SQL
Основы SQL Syntax
Group by, агрегационные функции
Оконные функции
Специализированные знания по некоторым языкам программирования
Java
Core
JVM
Maven
JDBC
Scala
Function programming
ZIO 2
Tagless final
Type classes
Akka
#C
.NET/.NET Core
Entity framework
NUnit /Allure
Task, async/await
WebAPI
Современные подходы к обработке Big Data и хранилища
noSQL
Hadoop
Column-based
K/V
Графовые
Документоориентированные
In-memory
Поисковые
DWH, Data Lake, Lake House
Шардирование, партиционирование, репликирование, сжатие, архивация
OLAP/OLTP

6. Заключение

Подход, изложенный в рамках данного документа, может использоваться при разработке, проектировании и оптимизации платформенных решений кибербезопасности в крупных предприятиях, а также позволит архитекторам и разработчикам сосредоточиться на решении существующих проблем, взяв за основу предложенную методологическую составляющую.

Дальнейшее развитие предложенного подхода по построению аналитических платформ кибербезопасности связано с применением методов интеллектуального анализа больших данных, прогнозирования кибератак, профилирования атакующих, выбора оптимальных контрмер с учетом точности и производительности.

СЛУЖБА
КИБЕРБЕЗОПАСНОСТИ

ДЕПАРТАМЕНТ ИТ БЛОКА
“СЕРВИСЫ” И БЕЗОПАСНОСТИ

Москва 2024

КОЛЛЕКТИВ АВТОРОВ:

Сергей
Кибершеф

Михаил
Главный руководитель разработки

Александр
Офицер безопасности

Михаил
Сервис потоковой загрузки

Иван
Сервис доступа к данными

Александр
Сервис управления метаданными

Иван
Сервис мониторинга

Валерий
Дизайнер

Александр Сергеевич
Владелец платформы

Алексей
Архитектор платформы

Евгений
Сервис поиска

Екатерина
Сервис пакетной загрузки

Максим
Сервис управления доступом

Сергей
Сервис Feature Store

Библиотека знаний
о кибербезопасности

Виртуальный тур по Центру
кибербезопасности СБЕРа