Программное обеспечение

«Программный комплекс Bot-Trek TDS»

Описание функциональных характеристик

Содержание

1 ОБЩИЕ СВЕДЕНИЯ

1.1 Введение

Настоящий документ содержит описание функциональных характеристик программного обеспечения «Программный комплекс Bot-Trek TDS» (далее – Bot-Trek TDS, ПО).

1.2 Назначение ПО

Программный комплекс Bot-Trek TDS является средством защиты конфиденциальной информации (системой обнаружения вторжений уровня сети) и предназначен для автоматизированного обнаружения компьютерных атак (вторжений) и вредоносного ПО в сетевом трафике при помощи сигнатурного метода выявления атак и эвристического анализа. Изделие устанавливается на границе сети с целью повышения уровня защищенности ИС, ЦОД, серверов и коммуникационного оборудования, АРМ пользователей.

2 ПРОГРАММНО-АППАРАТНЫЕ СРЕДЫ ФУНКЦИОНИРОВАНИЯ ПО

Изделие функционирует в среде операционной системы специального назначения «Astra Linux Special Edition» РУСБ.10015-01 (версия 1.6) Релиз «Смоленск» на отдельно выделенном сервере СОВ.

Для функционирования подсистемы аудита и базы данных системы обнаружения вторжений используется СУБД PostgreSQL версия 9.6 (входит в состав операционной системы специального назначения «Astra Linux Special Edition» РУСБ.10015-01 (версия 1.6) Релиз «Смоленск»).

Для хранения базы решающих правил используется файловая система ОС «Astra Linux Special Edition».

Для хранения событий, генерируемых сенсором, еще не попавших в базу, используется Redis – сетевое журналируемое хранилище данных типа «ключ - значение» (входит в состав операционной системы специального назначения «Astra Linux Special Edition» РУСБ.10015-01 (версия 1.6) Релиз «Смоленск»).

3 ОБЩИЕ ПРИНЦИПЫ ФУНКЦИОНИРОВАНИЯ ПО

Bot-Trek TDS обеспечивает следующие функциональные возможности:

- сбор информации о сетевом трафике;
- анализ собранных данных системы обнаружения вторжений о сетевом трафике в режиме, близком к реальному масштабу времени, и по результатам анализа фиксация информации о дате и времени, результате анализа, идентификаторе источника данных, протоколе, используемом для проведения вторжения;
- анализ собранных данных с целью обнаружения вторжений с использованием сигнатурного и эвристических методов;
- анализ собранных данных с целью обнаружения вторжений с использованием эвристических методов, основанных на методах выявления аномалий сетевого трафика на заданном уровне эвристического анализа;
- обнаружение вторжений на основе анализа служебной информации протоколов сетевого уровня базовой эталонной модели взаимосвязи открытых систем;
- фиксация факта обнаружения вторжений или нарушений безопасности в журналах аудита;
- уведомление администратора системы обнаружения вторжений об обнаруженных вторжениях по отношению к контролируемым узлам ИС и нарушениях безопасности с помощью отображения соответствующего сообщения (пиктограммы) в графическом интерфейсе;
- автоматизированное обновление базы решающих правил;
- тестирование (самотестирование) функций безопасности изделия (контроль целостности исполняемого кода изделия);
- управление режимом выполнения функций безопасности изделия со стороны уполномоченных администраторов (ролей);
- управление данными изделия со стороны уполномоченных администраторов (ролей);

- поддержка определенных ролей для изделия и их ассоциации с конкретными администраторами СОВ и пользователями ИС;
- администрирование изделия;
- генерация записей аудита для событий, потенциально подвергаемых аудиту;
- ассоциация каждого события аудита с идентификатором субъекта, его инициировавшего;
- предоставление возможности читать информацию из записей аудита;
- ограничение доступа к чтению записей аудита;
- поиск, сортировка, фильтрация данных аудита.

В основу функционирования сетевого сенсора (сетевого датчика) Bot-Trek TDS положен сигнатурный и эвристический метод выявления атак. Он обеспечивает обнаружение атак на основе специальных шаблонов (сигнатур) и эвристических правил, каждое из которых соответствует конкретной атаке. При получении исходных данных о сетевом трафике автоматизированной информационной системы Bot-Trek TDS производит их анализ на соответствие указанным шаблонам атак, имеющихся в базе данных.

В случае обнаружения сигнатуры или отработки правила в исходных данных изделие регистрирует факт обнаружения атаки, оповещает администратора безопасности о данном событии. За счет использования механизма контроля целостности Bot-Trek TDS позволяет отслеживать действия нарушителя по отношению к контролируемым ресурсам в скомпрометированной системе. Дополнительно поддерживается получение данных о функционировании отдельных объектов контролируемой системы по используемым приложениям в протоколах верхнего уровня (браузеры, почтовые клиенты).

Bot-Trek TDS реализует следующие методы реагирования на факт выявления компьютерной атаки:

- идентификация компьютерной атаки с использованием описаний уязвимостей, на которые они направлены, или описаний реализаций компьютерных атак;
- оповещение администратора безопасности об обнаруженных атаках;
- регистрация атаки в журнале аудита Bot-Trek TDS

Bot-Trek TDS имеет текстовую консоль, которая реализует механизм локального управления данным средством обнаружения атак и позволяющий: производить настройку своих компонентов, их запуск, остановку и перезапуск. Связь между удаленной текстовой консолью и компонентом изделия, выполняющим управление сетевым оборудованием, осуществляется по отдельно выделенному сетевому интерфейсу. Изделие позволяет в автоматическом режиме получать новые сообщения от датчиков для системных журналов контролируемой системы.

С целью маскирования изделия в составе контролируемой системы предполагается выделение изделия в отдельный сегмент, если на защищаемых объектах не установлены датчики контроля целостности, или отделение компонентов изделия от возможных нарушителей с помощью межсетевых экранов, исключая точки съема информации сетевыми датчиками. В качестве дополнительной меры по затруднению демаскирования компонентов изделия предусмотрена возможность наложения ограничений на сетевые адреса, между которыми осуществляется взаимодействие компонентов.

Bot-Trek TDS реализует следующие механизмы собственной защиты:

- обеспечивается идентификация и аутентификация администратора при запуске текстовой консоли по имени пользователя и паролю; ведется контроль длины создаваемых паролей (не менее 8 символов) и состав паролей (буквенно-цифровые);
- в процессе работы осуществляется контроль целостности компонентов и конфигурации изделия;
- имеет функцию сигнализации администратору безопасности о неверных попытках аутентификации при доступе к изделию, в частности, сигнализации о трех подряд неверных попытках аутентификации путем записи соответствующего события в системный журнал.

Изделие регистрирует в своих журналах аудита следующие события:

- сведения о выявленных компьютерных атаках и случаях нарушения целостности контролируемых ресурсов;
- сведения о сообщениях системных журналов с машин контролируемых ресурсов;

- служебную информацию, формируемую компонентами изделия, такую как подключение или отключение компонентов изделия, вход и выход администратора.

Дополнительные характеристики изделия:

- имеет механизм фильтрации событий, отображаемых в журналах;
- обладает интуитивно-понятным русскоязычным графическим интерфейсом и графическим текстовым интерфейсом администрирования;
- работает под управлением UNIX подобных операционных систем;
- обеспечивает анализ стека протоколов TCP/IP начиная с канального уровня;
- имеет возможность генерации табличных и текстовых отчетов на основе содержимого журналов;
- имеет распределенную модульную архитектуру, обеспечивающую масштабируемость системы, позволяющую адаптироваться под требования конкретной системы по производительности и отказоустойчивости;
- существует возможность резервирования ключевых компонентов.

4 РЕАЛИЗАЦИЯ ПО

Структура изделия приведена на рис. 1.

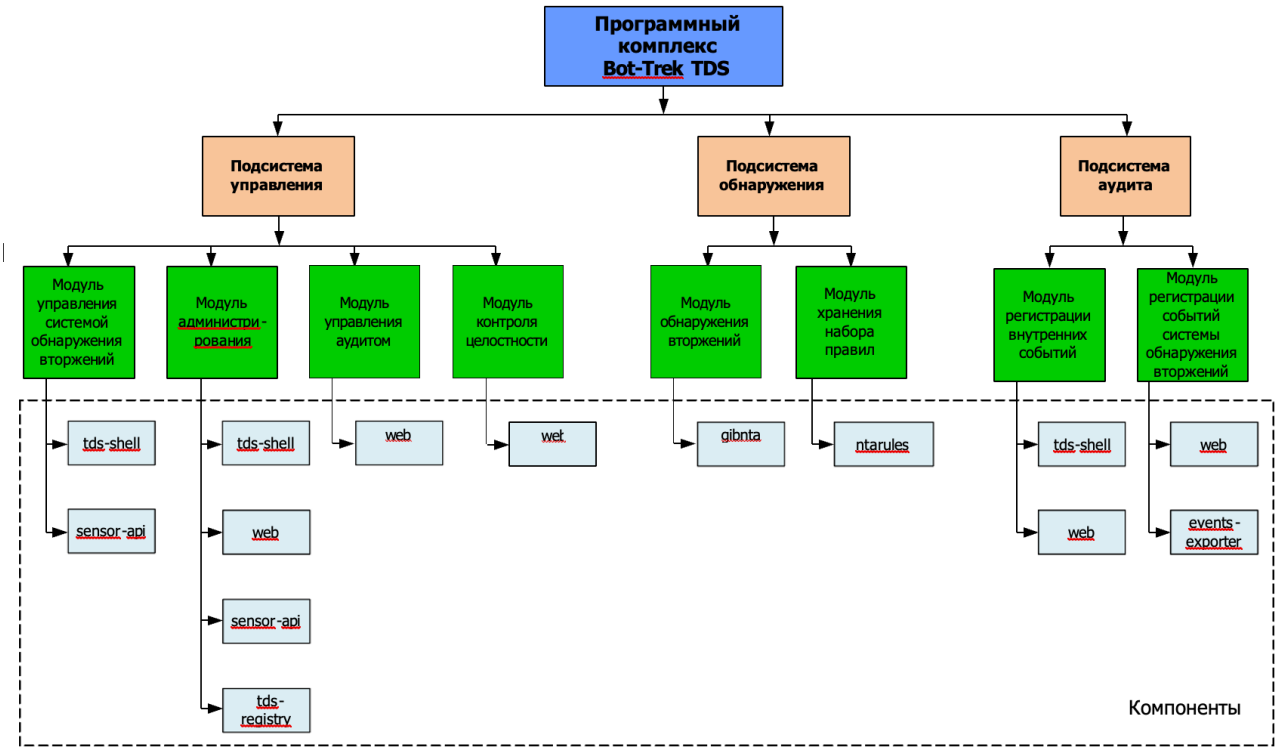

Рис. 1

Описание компонентов изделия приведен ниже:

1) **Gibnta** – сенсор, движок, выполняющий сигнатурный анализ.

2) **ntarules** – база решающих правил (БРП) − централизованное хранение набора правил, предназначенных для эвристического анализа трафика и набор сигнатур для выявления аномальных событий по атакам на защищаемые ресурсы.

3) **tds-shell** – консольная панель управления (текстовая консоль) с псевдографическим интерфейсом для первоначальной настройки.

4) **web** – web-интерфейс, панель администратора и аудит безопасности.

5) **sensor-api** – прослойка для tds-shell и web. Предоставляет им API для управления и настройки системы.

6) **events-exporter** – компонент «забирающий» события (данные) из Redis и «кладущий» их в базу данных (компонент events-exporter входит в состав компонента web).

7) **tds-registry** – конфигурационный файл и механизм управления им для работы системы в целом.

4.1 Работа системы обнаружения вторжений

Работа системы обнаружения вторжений неизменна для всех типов детектируемых угроз.

Описание работы системы обнаружения вторжения вторжений приведено на Рис.2.

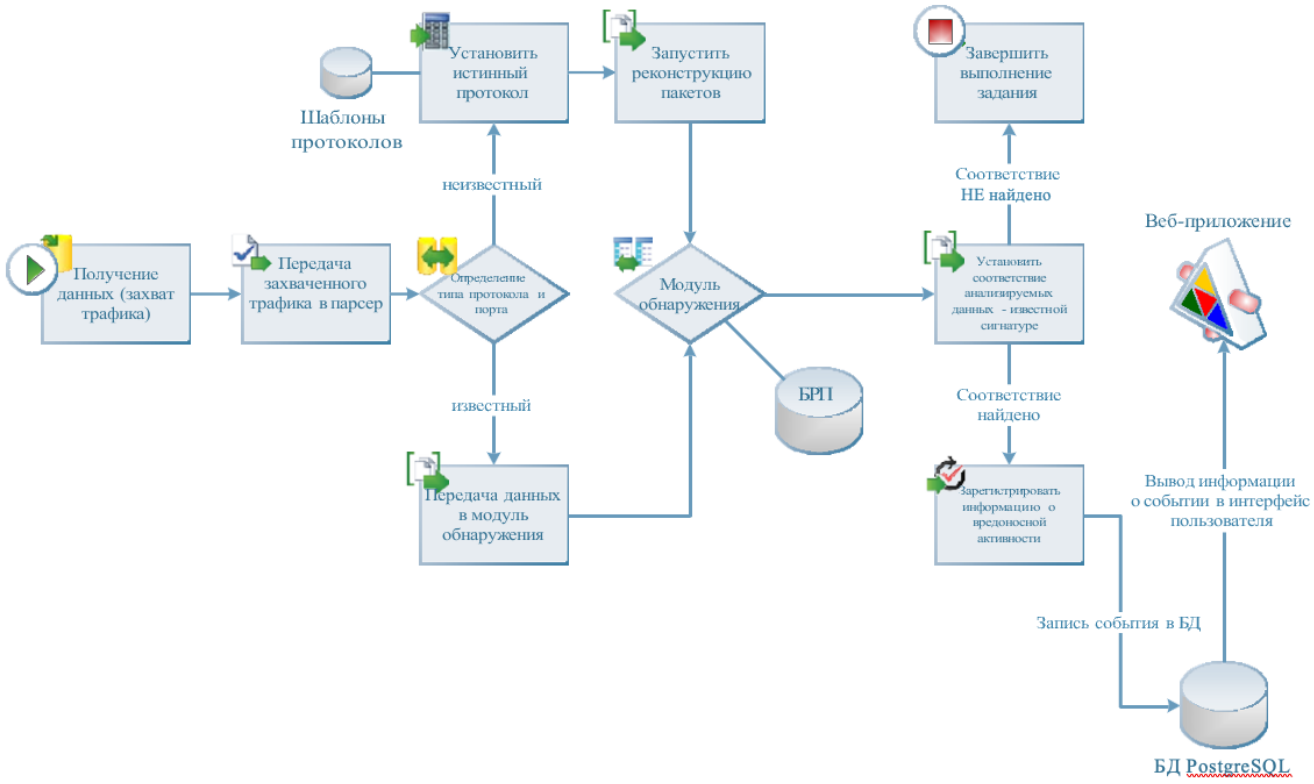

Рис. 2

Система обнаружения вторжений действует по приведенному алгоритму, то есть анализирует поток поступающего на ее вход трафика, при этом входящий трафик разбивается на TCP, UDP или другие транспортные потоки, после чего парсеры пакетов (синтаксические анализаторы) маркируют их и разбивают на высокоуровневые протоколы и их поля − нормализуя, если требуется. Полученные декодированные, раскрытые и нормализованные поля протоколов (пакеты) затем проверяются наборами сигнатур,

которые выявляют есть ли среди сетевого трафика попытки сетевых атак или пакеты, присущие вредоносной активности.

Работа модуля обнаружения приведена на рис. 3.

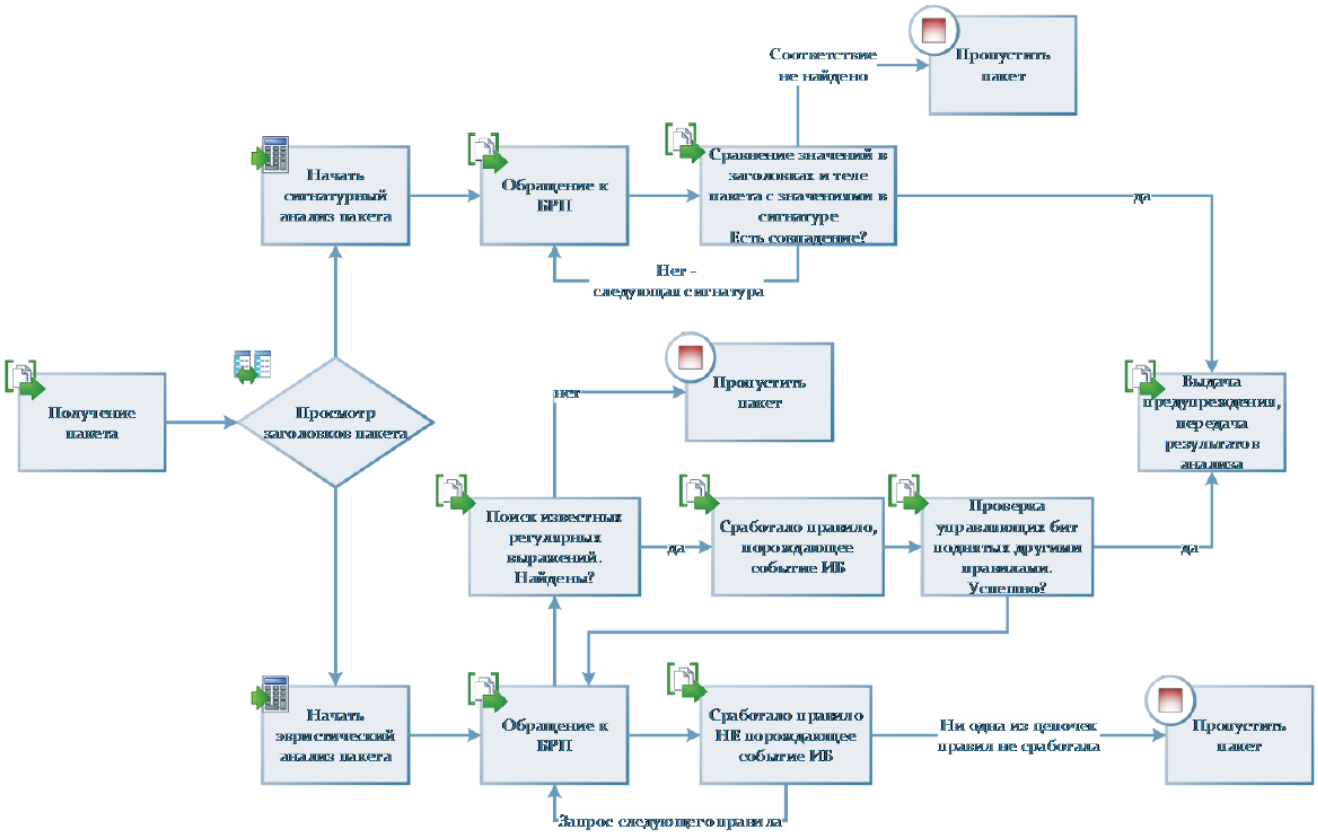

Рис. 3

С получением на вход пакета модуль обнаружения начинает его обработку. Анализируются заголовки пакета, запускаются процессы сигнатурного и эвристического анализа. Сигнатурный анализ характеризуется непрерывным сравнением значений в заголовках и теле пакета с значениями, указанными в сигнатуре. Модуль обнаружения в этом случае будет перебирать сигнатуры из Базы решающих правил до тех пор, пока не будет выявлено соответствие известной сигнатуре, которая относится к тому, или иному классу. Если такого соответствия установлено не будет, модуль пропустит пакет и его анализ на этом завершится.

Анализ пакета эвристическим методом так же начинается с выявления соответствия значений в заголовках и теле пакета наборам эвристических правил, хранящихся в Базе решающих правил. Основным отличием данного метода от сигнатурного является сопоставление значений в заголовках пакета не конкретным сигнатурам, а цепочкам правил, которые делятся на два типа:

– порождающие события;

– не порождающие события.

Каждое из правил может как поднимать управляющие биты (флаги) для анализируемой информации, так и проверять наличие уже поднятых другими правилами флагов.

При этом пакет может пройти проверку несколькими правилами, которые не порождают событий ИБ. В этом случае событие информационной безопасности не будет зафиксировано модулем обнаружения и анализ пакета эвристическим методом завершится. Если в цепочке правил сработало правило, порождающее событие ИБ, модуль обнаружения производит сопоставление значений в теле пакета известным ему регулярным выражениям и проверяет всю цепочку правил, которые поднимали управляющие биты. При этом управляющие биты могли быть подняты в ходе анализа предыдущих сетевых пакетов.

Результатом работы модуля обнаружения в этом случае является выдача предупреждения о событии ИБ и передача данных о сработавших правилах в базу данных.

Такой подход к анализу сетевого трафика позволяет существенно повысить вероятность обнаружения атак (вторжений), распределенных во времени, и в том случае если эксплуатация уязвимости реализуется распределением тела атаки в разных сетевых пакетах хаотично и непоследовательно.

Таким образом модуль обнаружения, используя эвристический метод анализа сетевого трафика, опирается на Базу решающих правил в принятии решения о выдаче предупреждения и регистрации событий информационной безопасности.

На рис. 4 показана последовательность действий при разборе сетевого пакета:

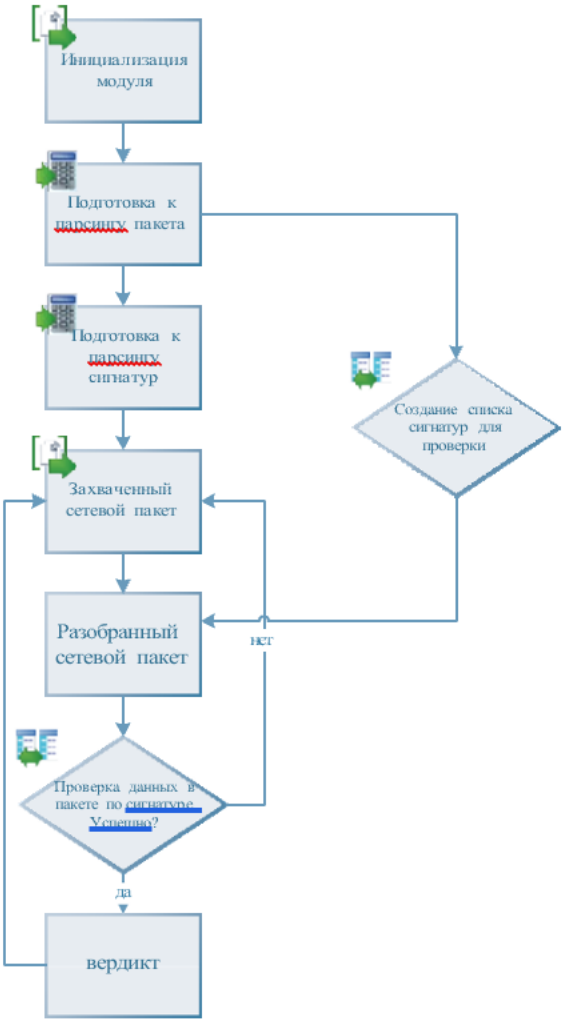

Рис. 4

Каждый сетевой пакет разбирается отдельно, данные из пакета перед проведением анализа сохраняются в структурированном виде. Эта структура с сохраненной информацией, извлеченной из захваченного пакета данных, необходима для последующей процедуры проверки указанных данных по сигнатурам и цепочкам правил.

После завершения парсинга пакетов модуль обнаружения сравнивает результаты анализа с сигнатурами и цепочками правил, чтобы определить, произошло ли вторжение, если будет установлено соответствие, модуль обнаружения вынесет вердикт с указанием сработавшей сигнатуры или цепочки правил.

Успешность сигнатурного и эвристического метода анализа трафика полностью зависит от количества и качества загруженных сигнатур и правил в базу решающих правил. База регулярно обновляется, сигнатуры и правила ежемесячно дополняются, но при этом структура системы обнаружения вторжений и схема ее работы не изменяются.

База решающих правил состоит из множества сигнатур. Сигнатуры по своей сути − это набор данных, на которые опирается сенсор, анализируя трафик. Этот набор данных в большинстве своем берется из тела атаки (файла вредоносного ПО или сетевого пакета, принадлежащего эксплойту). Поэтому классификация сигнатур представляет из себя виды угроз, реализуемые с помощью того или иного вида вредоносного ПО или метода реализации вторжения (атаки) на защищаемую сетевую инфраструктуру.

Выполнение поставленных задач достигается за счет:

- наличия облачного интерфейса – вся информация о выявленных угрозах доступна в веб-интерфейсе, через который удобно отслеживать уведомления в течение дня;

- применения наглядных отчетов – визуализированная статистика по периодам и по типам событий позволяет отслеживать изменения в динамике и характере атак.

4.2 Список использованных при разработке ПО сторонних компонентов

При разработке и тестировании программного обеспечения использовался следующий инструментарий:

- компилятор GCC для языков C, C++.

- компилятор g++ языка C++.

- IDE SublimeText с набором плагинов. Плагины включают в себя подсветку и проверку синтаксиса. Синхронизация с системой контроля версия Git.

- текстовый редактор Sublime Text. Поддерживает плагины на языке программирования Python.

- SonarQube — платформа с открытым исходным кодом для непрерывного анализа и измерения качества кода.

- Cppcheck — статический анализатор кода для языка C/C++, предназначенный для поиска ошибок, которые не обнаруживаются компиляторами. Главной целью проекта является сведение до минимума количества ложных срабатываний при поиске ошибок.

- Eslint — это инструмент, который позволяет проводить анализ качества кода, написанного на любом выбранном стандарте JavaScript. Он приводит код к

единому стилю, помогает избежать ошибок, умеет автоматически исправлять многие из найденных проблем и хорошо интегрируется со многими инструментами разработки. Для проведения статического анализа JavaScript в составе Eslint использовался плагин ScanJS. который был создан в качестве вспомогательного средства для проверки, чтобы помочь выявить проблемы безопасности в клиентских веб-приложениях. ScanJS использует Acorn - небольшой, быстрый парсер JavaScript, написанный полностью на JavaScript.

5 ВХОДНЫЕ И ВЫХОДНЫЕ ДАННЫЕ

Входными данными ПО являются:

Данные от модулей системы и управляющие команды пользователей.

Выходными данными ПО является:

Проанализированная информация от модулей системы, которая разбивается на определенные группы. С их помощью проводится мониторинг, реагирование на инциденты и проведение расследований в защищаемой инфраструктуре.