МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»
Кафедра Систем информационной безопасности

«ДОПУЩЕН К ЗАЩИТЕ»
Зав. кафедрой к.п.н., доцент Р. Ш. Бердибаев
_____ «_____» _____2019 г.
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Защита серверов на Amazon Web Services
Специальность Системы информационной безопасности
Выполнил Амангелдин Баян Батырулы Группа СИБ-15-2
Научный руководитель Ургенишбаев Камал Махамбетович

Консультант:
по экономической части:
к.э.н., профессор Аренбаева Н.Г.
(ученая степень, звание, Ф.И.О)
(подпись) «24» мая 2019 г.

по безопасности жизнедеятельности:
ст. пр. Бекбасаров И.И.
(ученая степень, звание, Ф.И.О)
(подпись) «23» мая 2019 г.

по применению вычислительной техники:
ст. пр. Ургенишбаев К.М.
(ученая степень, звание, Ф.И.О)
(подпись) «23» мая 2019 г.

Нормоконтролер:
ст. преподаватель Аскарова Н.Т.
(ученая степень, звание, Ф.И.О)
(подпись) «24» 05 2019 г.

Рецензент:
Ахметов Сарсен Бимуратович
(ученая степень, звание, Ф.И.О)
(подпись) «23» мая 2019 г.

Алматы 2019

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН
Некоммерческое акционерное общество
«АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ»

Институт систем управления и информационных технологий

Кафедра Систем информационной безопасности

Специальность Системы информационной безопасности

ЗАДАНИЕ
на выполнение дипломного проекта

Студенту Амангелдину Баяну Батырулы

Тема проекта Защита серверов на Amazon Web Services

Утверждена приказом по университету №_124_ от «_26_»_октября_2018 г.

Срок сдачи законченного проекта «_____» _____2019 г.

Исходные данные к проекту (требуемые параметры результатов исследования (проектирования) и исходные данные объекта): проект предполагает развертывание и обеспечение защиты WEB и почтовых серверов на Amazon Web Services. В качестве системы защиты будут использоваться: группы безопасности Amazon Web Services, WAFnaxsi, Cloudflare CDN, Защита от перебора парлей, SPF, DKIM, DMARC, SpamAssassin, ClamAV.

Перечень вопросов, подлежащих разработке в дипломном проекте, или краткое содержание дипломного проекта: дипломный проект включает в себя 5 глав, разделенных на подглавы, каждая из которых освещает определенную тематику, используемую при защите серверов на Amazon Web Services.

В первой главе дипломного проекта представлена общая информация по облачным вычислениям: определение облачных вычислений, основные характеристики, сервисные модели, модели развертывания. Подробно описан сервис Amazon EC2.

Во второй главе дипломного проекта произведен анализ веб-серверов: виды, доля на рынке, механизмы работы, поддерживаемые функций; и меры защиты серверов.

В третьей главе подробно описан весь процесс разработки, подкрепленный скриншотами.

Четвертая глава посвящена экономической составляющей дипломного проекта, в которой приведено технико-экономическое обоснование, а также все необходимые расчеты.

Пятая глава посвящена расчетам, необходимым для создания вентиляционных условий в помещении, пригодных к комфортной работе специалистов.

Перечень графического материала (с точным указанием обязательных чертежей):

1 принципы работы необходимых составляющих для защиты серверов;

2 скриншоты конфигурационных файлов;

3 скриншоты интерфейсов средств защиты;

4 скриншоты результатов сканирования WEB-сервера;

5 скриншоты результатов тестирования безопасности.

Основная рекомендуемая литература:

1 Денисов Д.В. Перспективы развития облачных вычислений. – М.: Издательский дом Университета "Синергия", 2009

2 Крис Б. Безопасность Linux сервера. Атака и защита. – «John Wiley and Sons, Inc»:2016.

3 Дэйв В. Защита от хакеров Web-приложений. – «Айти , ДМК Пресс»:2018.

4 Хакимжанов Т.Е. Расчет аспирационных систем. Дипломное проектирование. Для студентов всех форм обучения всех специальностей. – Алматы: АУЭС, 2014.

5 Бекишева А.И. Методические указания к выполнению экономической части дипломной работы для бакалавров специальности 5В0703 – Информационные системы – Алматы: АУЭС, 2013.

Конструкции по проекту с указанием относящихся к ним разделов проекта

Раздел	Консультант	Сроки	Подпись
БЖД	Бекбасаров Ш.Ш	04.03-23.05.19	
Экономика	Аренбаева Ж.Г.	04.03-24.05.19	
	Утешимбаев К.М.		
рецензия	Бакулиев С		
Нормоконтроль	Аскарова Н.Б		

График
подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	Сроки представления научному руководителю	Примечание
Определение актуальности темы	22.10.2018	
Анализ предметной области	12.11.2018	
Теоретический материал о облачном ПО	04.02.2019	
Теоретический материал о web- и почт. серв.	13.02.2019	
Анализ рынка web-серверов. Их сравнение	22.02.2019	
Определение мер защиты сервера	05.03.2019	
Развертывание виртуального сервера Amazon EC2	12.03.2019	
Настройка группы безопасности AWS	13.03.2019	
Развертывание web-сервера	18.03.2019	
Установка и настройка WAF	20.03.2019	
Обеспечение защиты от распространенных атак	25.03.2019	
Установка и настройка LEMP и Wordpress	01.04.2019	
Обеспечение безопасности средствами Wordpress	05.04.2019	
Тестирование безопасности сканерами	11.04.2019	
Устранение "брешей" в безопасности	15.04.2019	
Развертывание почтового сервера	22.04.2019	
Настройка SPF, DKIM, DMARC	25.04.2019	
Настройка Postfix для защиты от спама	06.05.2019	
Настройка Spamassassin для защиты от спама	06.05.2019	
Анализ проделанной работы	15.05.2019	

Дата выдачи задания «___» _____2019 г.

Заведующий кафедрой _____ (_____)
(Подпись) (Ф.И.О)

Научный руководитель проекта _____ (Ургешбаев К.М.)
(Подпись) (Ф.И.О)

Задание принял к исполнению студент _____ (Алмагелди Б.Б)
(Подпись) (Ф.И.О)

Аннотация

Данный дипломный проект посвящен изучению предметной области облачных платформ, в особенности виртуальных серверов EC2, предоставляемые компанией Amazon на их собственной облачной платформе Amazon Web Services – AWS, изучение предметной области WEB и почтовых серверов и их защите на виртуальных серверах EC2.

В экономической части произведен расчет возможных затрат при разработке дипломного проекта.

В разделе безопасности жизнедеятельности определены оптимальные условии труда, произведен расчет освещенности и расчет для системы вентиляции в помещении.

Андатпа

Бұл дипломдық жоба бұлтты платформалар, әсіресе Amazon компаниясының Amazon Web Services – AWS өзінің бұлтты платформасында ұсынатын EC2 виртуалды серверлерінің пәндік саласын зерттеуге, WEB және пошта серверлерінің пәндік саласын зерттеуге және оларды EC2 виртуалды серверлерінде қорғауға арналған.

В экономической части произведен расчет возможных затрат при разработке дипломного проекта.

Өміртіршілік қауіпсіздігі бөлімінде тиімді еңбек шарттары анықталған, жарықтандыруды есептеу және бөлмедегі желдету жүйесі үшін есептеу жүргізілген.

Annotation

This graduation project is devoted to the study of the subject area of cloud platforms, especially virtual EC2 servers provided by Amazon on their own cloud platform Amazon Web Services – AWS, the study of the subject area of WEB and mail servers and their protection on virtual EC2 servers.

In the economic part of the calculation of possible costs in the development of the diploma project.

In the section of life safety optimum working conditions are defined, calculation of illumination and calculation for system of ventilation indoors is made.

Содержание

Введение

Облачные вычисления, пожалуй, самая яркая технологическая инновация XXI века. Это связано с тем, что развитие этой технологии, а точнее модели, напрямую связано с развитием Интернета. А в свою очередь, потребность в Интернете, с каждым днём всё глубже и глубже проникающий в различные сферы человеческой жизни, вызвал резкий рост числа мобильных устройств, подключенных к нему. Облачные вычисления предназначены не только для организаций и предприятий; модель также полезна для обычных пользователей. Она может позволить нам запускать программы без их установки на наших компьютерах, хранить и получать доступ к нашему мультимедийному контенту через Интернет, разрабатывать и тестировать программы без необходимости иметь серверы и так далее.

Доказано, что объединенные вычислительные ресурсы, доступные через облачные вычисления, дают огромные преимущества для бизнес-организаций. Эти преимущества раскрываются в трех категориях: эффективность, гибкость и стратегическая ценность.

Эффективность бизнес-операций достигается с помощью облачных вычислений следующими способами:

- **доступность** – облачные вычисления облегчают доступ к приложениям и данным из любой точки мира и с любого устройства, подключенного к Интернету;
- **экономия на издержках** – облачные вычисления предлагают предприятиям масштабируемые вычислительные ресурсы, что позволяет им сэкономить на затратах на их приобретение и обслуживание. Эти ресурсы оплачиваются по мере поступления, что означает, что предприятия платят только за ресурсы, которые они используют. Это оказалось гораздо дешевле, чем приобретать ресурсы самостоятельно;
- **безопасность** – облачные провайдеры, особенно те, которые предлагают частные облачные сервисы, стремились внедрить лучшие стандарты и процедуры безопасности для защиты данных клиента, сохраненных в облаке;
- **аварийное восстановление** – облачные вычисления предлагают наиболее эффективные средства для малых, средних и даже крупных предприятий для быстрого и надежного резервного копирования и восстановления своих данных и приложений;

При использовании облачных вычислений гибкость достигается следующими способами:

- **масштабируемость** – облачные вычисления – лучший вариант для предприятий с колеблющимися рабочими нагрузками, поскольку облачная инфраструктура масштабируется в зависимости от потребностей бизнеса;
- **выбор инструментов** – облачные вычисления позволяют компаниям выбирать конкретные готовые инструменты и функции для выработки решений, адаптированных к их конкретным потребностям;

- **облачные опции** – облачные вычисления предлагают частное облако, общедоступное облако и гибридные облачные решения, каждое из которых имеет различные функции. Организации могут выбирать эти варианты в зависимости от того, что лучше всего соответствует их потребностям;
- **контроль выбора** – предприятия могут определять свой уровень контроля с помощью опций «как услуга», предлагаемых облачным провайдером. Эти варианты включают SaaS, PaaS и IaaS;

Облачные вычисления позволяют компаниям получить стратегическое преимущество в своей нише следующими способами:

- **повышение производительности труда** – поставщики облачных услуг приобретают и управляют базовой облачной инфраструктурой, что позволяет компаниям сосредоточить свои усилия на своей основной деятельности;
- **автоматическое обновление программного обеспечения** – все программные приложения, доступ к которым осуществляется через облако, как правило, обновлены. Это позволяет предприятиям получать доступ к новейшим функциям без необходимости поддерживать систему самостоятельно;
- **конкурентоспособность** – предприятия, которые используют облачные вычисления, способны маневрировать более ловко по сравнению с конкурентами, которые посвящают свою энергию приобретению и поддержанию ИТ-ресурсов;
- **расширение сотрудничества** – обладая возможностями облачных вычислений, люди из разных мест могут сотрудничать в бизнес-проектах без необходимости встречаться;

Актуальность моей работы заключается в том, что всё больше и больше компаний используют облачные платформы для решения разного рода задач, например, используют виртуальные облачные серверы в качестве WEB и почтовых серверов. Эти сервера являются крайне, зачастую, самыми ценными информационными активами. Следовательно, они нуждаются в защите от всевозможных атак.

Целью данного дипломного проекта является развертывание и Защита WEB и почтовых серверов на базе Linux-сервера на платформе Amazon Web Services.

Задачи, которые я поставил для достижения своей цели:

- изучение теоретического материала на тему облачных вычислений, WEB и почтовых серверов;
- развертывание и настройка WEB и почтовых серверов на платформе Amazon Web Services;
- устранение “брешей” в безопасности серверов;
- рассмотрение БЖД и экономической части;
- анализ выполненной работы.

1 Анализ предметной области

1.1 Основные понятия и определения облачных вычислений

Облачные вычисления – это предоставление вычислительных мощностей, хранилищ для БД, приложений и других ИТ-ресурсов по требованию на платформах облачных сервисов через Интернет с оплатой по факту использования.

Платформа облачных сервисов предоставляет быстрый доступ к гибким и недорогим ИТ-ресурсам, необходимым как при запуске приложений для публикации фотографий миллионов пользователей, так и при управлении важнейшими бизнес-процессами в компании. Облачные вычисления позволяют избавиться от больших предварительных затрат на оборудование и сэкономить время, необходимое для трудоемкого управления им. Вместо этого можно распределить вычислительные ресурсы таких типов и размеров, которые необходимы для реализации ваших новых ярких идей или для управления ИТ-отделом. Можно практически мгновенно получать доступ к необходимому количеству ресурсов с оплатой по факту использования [1].

Облачные вычисления обеспечивают простой доступ к серверам, хранилищам, базам данных и обширному набору сервисов приложений в Интернете. Платформы облачных сервисов, такие как Amazon Web Services, владеют подключенным к сети оборудованием, необходимым для подобных сервисов приложений, и выполняют его обслуживание, в то время как вы распределяете и используете необходимые ресурсы через интернет-приложение.

1.2 Определение облачных вычислений NIST

Национальный институт стандартов и технологий (NIST) одним из первых попытался дать определение облачным вычислениям в специальной публикации 800-145: «Облачные вычисления это модель предоставления повсеместного и удобного сетевого доступа «по мере необходимости» к общему пулу конфигурируемых вычислительных ресурсов (например, сетей, серверов, систем хранения, приложений и сервисов), которые могут быть быстро предоставлены и освобождены с минимальными усилиями по управлению и необходимостью взаимодействия с провайдером услуг (сервис-провайдером)».

Облачная модель поддерживает высокую доступность сервисов и описывается пятью основными характеристиками (essential characteristics), тремя сервисными моделями/моделями предоставления услуг (service models) и четырьмя моделями развертывания (deployment models)[3].

Основные характеристики (Essential Characteristics):

- сервис самообслуживания по необходимости или по требованию (On-demand self-service) – потребитель может самостоятельно обеспечивать себя вычислительными возможностями (средствами и ресурсами), такими как

серверное время и сетевое хранилище, по мере необходимости автоматически, без необходимости взаимодействие с каждым поставщиком услуг;

- свободный доступ к сети (Broad network access) – вычислительные возможности или "запрашиваемые" сервисы доступны по сети через стандартные механизмы, поддерживающие использование гетерогенных платформ тонких и толстых клиентов (например, мобильных телефонов, ноутбуков и КПК);
- объединение ресурсов (Resource pooling) – вычислительные ресурсы провайдера объединяются для обслуживания нескольких потребителей с использованием множественной модели (Multi-tenant model - модель множественной аренды или, точнее, модель аренды с множеством арендаторов - предполагает доступность потребителям специфического для каждого из них портфеля и объема ресурсов из общего спектра ресурсов, поддерживаемых провайдером) с различными физическими и виртуальными ресурсами, которые динамически назначаются и переназначаются в соответствии с требованиями потребителей. Существует ощущение независимости местоположения в том, что клиент, как правило, не имеет никакого контроля или знаний о точном местоположении предоставленных ресурсов, но может иметь возможность указать местоположение на более высоком уровне абстракции (например, страна, штат или центр обработки данных). Примерами таких ресурсов являются системы хранения, вычислительные возможности, память, пропускная способность сети, виртуальные машины;
- быстрая эластичность (Rapid elasticity) – вычислительные возможности могут быть предоставлены быстро и эластично, в ряде случаев - автоматически, для оперативного повышения масштабируемости и быстрого освобождения для уменьшения масштабов потребления . Для потребителя эти ресурсы часто представляются, как доступные в неограниченном объеме, и могут быть приобретены в любой момент времени в любом количестве;
- измеримый сервис (Measured Service) – облачные системы автоматически контролируют и оптимизируют использование ресурсов, используя возможности измерения на некотором уровне абстракции, соответствующем типу услуги (например, хранение, обработка, пропускная способность и активные учетные записи пользователей). Использование ресурсов можно отслеживать, контролировать и отчитываться, обеспечивая прозрачность как для поставщика, так и для потребителя используемой услуги.

Сервисные модели (Service Models):

- программное обеспечение как услуга (SaaS) – потребителю предоставляется возможность использовать приложения провайдера, работающие в облачной инфраструктуре. Приложения доступны с различных клиентских устройств через интерфейс клиента, такой как в WEB-браузер (например, электронная почта через Интернет), или программный интерфейс. Потребитель не управляет и не контролирует базовую облачную

инфраструктуру, включая сеть, серверы, операционные системы, хранилище или даже отдельные функции репликации, за возможным исключением ограниченных параметров конфигурации приложения;

- платформа как услуга (PaaS) – потребителю предоставляется возможность развертывания в облачной инфраструктуре созданных потребителем или приобретенных приложений, созданных с использованием языков программирования, библиотек, служб и инструментов, поддерживаемых поставщиком. Потребитель не управляет и не контролирует базовую облачную инфраструктуру, включая сеть, серверы, операционные системы или хранилище, но контролирует развернутые приложения и, возможно, параметры конфигурации для среды размещения приложении;
- инфраструктура как услуга (IaaS) – возможность, предоставляемая потребителю, заключается в обеспечении обработки, хранения, сетей и других основных вычислительных ресурсов, где потребитель может развертывать и запускать произвольное программное обеспечение, которое может включать в себя операционные системы и приложения. Потребитель не управляет и не контролирует базовую облачную инфраструктуру, но контролирует операционные системы, хранилище и развернутые приложения; и, возможно, ограниченное управление отдельными сетевыми компонентами (например, брандмауэрами хоста).

Модели развертывания (Deployment Models):

- частное облако (Private cloud) – облачная инфраструктура предоставляется для исключительного использования одной организацией, состоящей из нескольких потребителей (например, бизнес-единиц). Оно может принадлежать, управляться и эксплуатироваться организацией, третьей стороной или какой-либо их комбинацией, а также может существовать в помещении или за его пределами;
- облако сообщества (Community cloud) – облачная инфраструктура предоставлена для исключительного использования определенным сообществом потребителей из организаций, которые разделяют общие принципы (например, миссия, требования безопасности, политика и соображения соответствия). Оно может принадлежать, управляться и эксплуатироваться одной или несколькими организациями в сообществе, третьей стороной или какой-либо их комбинацией, а также может существовать в помещении или за его пределами;
- публичное облако (Public cloud) – облачная инфраструктура предназначена для открытого использования широкой публикой. Оно может принадлежать, управляться и эксплуатироваться коммерческой, академической или государственной организацией или какой-либо их комбинацией. Оно существует на территории облачного провайдера;
- гибридное облако (Hybrid cloud) – облачная инфраструктура представляет собой совокупность двух или более отдельных облачных инфраструктур (частной, общественной или общедоступной), которые остаются уникальными объектами, но связаны друг с другом

стандартизированной или частной технологией, которая обеспечивает портируемость данных и приложений (например, разрыв облака для балансировки нагрузки между облака) [3].

1.3 Виртуальные серверы от Amazon

EC2 – самый известный сервис, предоставляемый Amazon Web Services, представляющий собой виртуальные серверы основе Windows и UNIX. Amazon Elastic Compute Cloud (Amazon EC2) обеспечивает масштабируемую вычислительную мощность в облаке Amazon Web Services (AWS). Amazon EC2 позволяет увеличивать или уменьшать вычислительную мощность за несколько минут, а не часов или дней. Можно вводить в эксплуатацию один серверный инстанс, сотни или даже тысячи серверных инстансов одновременно. Можно также использовать Amazon EC2 Auto Scaling, чтобы поддерживать доступность группы инстансов EC2 и автоматически масштабировать группу в нужном направлении в зависимости от потребностей, чтобы поддерживать максимальную производительность и сокращать затраты[2].

Amazon EC2 размещается в нескольких местах по всему миру. Эти местоположения состоят из регионов и зон доступности. Каждый регион является отдельной географической зоной. Каждый регион имеет несколько изолированных мест, известных как зоны доступности. Каждая зона доступности изолирована, но зоны доступности в регионе связаны через каналы с низкой задержкой. Amazon EC2 предоставляет возможность размещать ресурсы, такие как экземпляры, и данные в нескольких местах. Ресурсы не копируются по регионам, если вы не сделаете это специально. Рисунок 1.1 иллюстрирует взаимосвязь между регионами и зонами доступности:

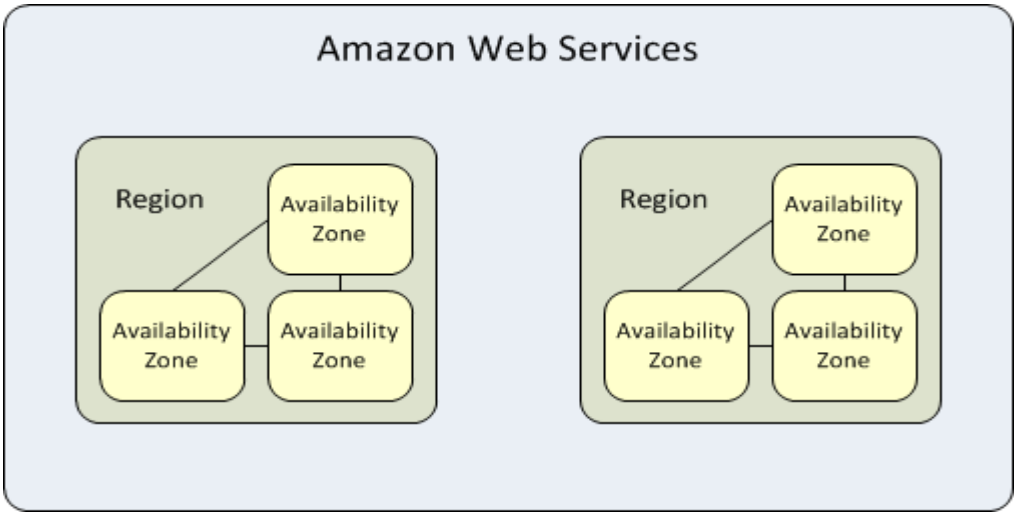

Рисунок 1.1 – Взаимосвязь между регионами и зонами доступности

В Amazon EC2 инстансами называют виртуальные серверы. Инстансы T2 – это инстансы с повышаемой производительностью, которые обеспечивают базовый уровень производительности ЦПУ с возможностью его повышения. Они могут поддерживать высокую производительность ЦПУ до тех пор, пока этого требует рабочая нагрузка. Для большинства рабочих нагрузок общего назначения безлимитные инстансы T2 обеспечивают достаточную производительность без дополнительной платы.

Особенности инстанстов T2:

- высокочастотные процессоры Intel Xeon;
- постоянный базовый уровень производительности и ЦПУ с возможностью повышения производительности (регулируется кредитами ЦПУ);
- самый экономичный тип инстансов общего назначения. Доступен на уровне бесплатного пользования;
- сбалансированное соотношение вычислительных, сетевых ресурсов и ресурсов памяти.

Для бесплатного пользования доступен инстанс T2 со следующими характеристиками:

- 1 виртуальный процессор Intel Xeon Family 2,5 ГГц;
- 1 ГБ оперативной памяти;
- 8 Гб постоянной памяти.

Группы безопасности AWS (security groups) связаны с экземплярами EC2 и обеспечивают безопасность на уровне протокола и доступа к порту. Каждая группа безопасности, работающая почти так же, как и брандмауэр, содержит набор правил, которые фильтруют трафик, поступающий в экземпляр EC2 и выходящий из него. Здесь нет никаких правил «Запретить». Скорее, если не существует правила, которое явно разрешает определенный пакет данных, он будет отброшен. Изменить правила для группы безопасности можно в любое время; новые правила автоматически применяются ко всем инстансам, которые связаны с группой безопасности.

Для безопасного входа в систему Amazon EC2 использует криптографию с открытым ключом для шифрования и дешифрования информации. Криптография с открытым ключом использует открытый ключ для шифрования фрагмента данных, а затем получатель использует закрытый ключ для дешифрования данных. Открытый и закрытый ключи известны как пара ключей. Криптография с открытым ключом позволяет безопасно получать доступ к инстансам, используя закрытый ключ вместо пароля.

При создании инстанса, нужно указать пару ключей. Можно указать существующую пару ключей или создать новую пару ключей. Во время загрузки содержимое открытого ключа помещается в экземпляр в записи в ~ / .ssh / authorized_keys. Чтобы войти в систему необходимо указать закрытый ключ при подключении к экземпляру [1].

Amazon Elastic Block Store (Amazon EBS) предоставляет тома хранилищ на уровне блоков для использования с экземплярами EC2. Тома EBS - это

высокодоступные и надежные тома хранения, которые можно подключить к любому работающему экземпляру в той же зоне доступности. Тома EBS, подключенные к экземпляру EC2, отображаются как тома хранения, которые сохраняются независимо от срока службы экземпляра.

На рисунке 1.2 видно, как происходит подключение к инстансу:

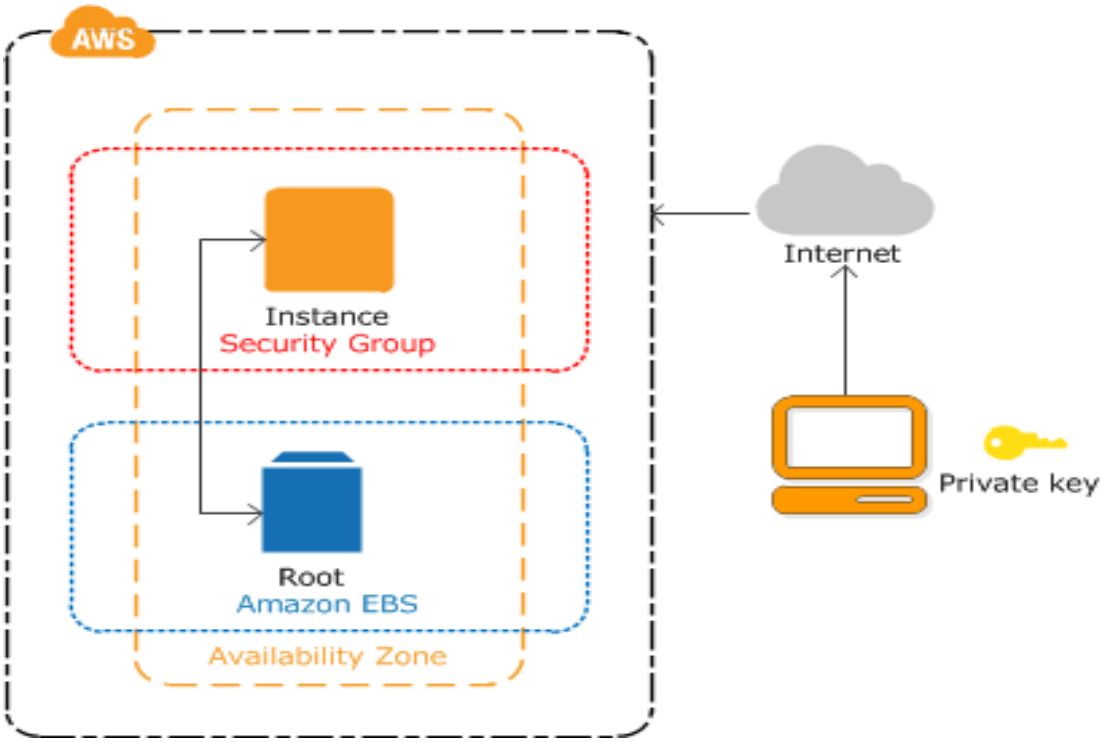

Рисунок 1.2 – Схема подключения к инстансу EC2

1.4 WEB-сервер и почтовый сервер

WEB-сервер – это компьютер, на котором хранится WEB-контент. В основном WEB-сервер используется для размещения WEB-сайтов, но существуют и другие WEB-серверы, такие как игровые сервера, хранилище, FTP и т. д. Лучшим определением может быть то, что WEB-сервер - это сервис, который отвечает на HTTP-запросы на доставку контента и услуг.

WEB-сервер отвечает на запрос клиента одним из следующих двух способов:

- отправка файла клиенту, связанному с запрошенным URL;
- генерация ответа путем вызова скрипта и связи с базой данных.

На рисунке 1.3 иллюстрировано, каким образом происходит взаимодействие WEB-клиента и WEB-сервера.

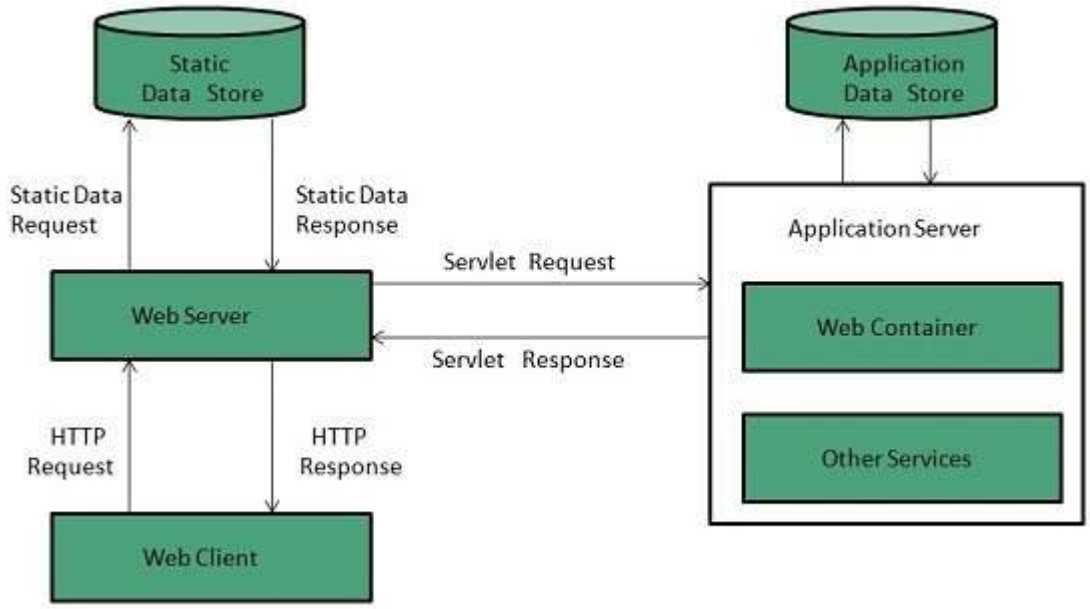

Рисунок 1.3 – Архитектура взаимодействия WEB-клиента и WEB-сервера

Архитектура WEB-сервера использует следующих два подхода для обработки запросов:

- параллельный подход;
- подход, основанный на отдельных процессах.

Параллельный подход позволяет WEB-серверу обрабатывать несколько клиентских запросов одновременно. Это может быть достигнуто следующими методами:

- мульти-процессный метод;
- многопоточный метод;
- гибридный метод.

При использовании мульти-процессного метода родительский процесс инициирует несколько однопоточных дочерних процессов и распределяет входящие запросы этим дочерним процессам. Каждый из дочерних процессов отвечает за обработку одного процесса.

Ответственность за контролем нагрузки и принятие решения о том, следует ли уничтожать или разветвлять процессы, лежит на родительском процессе.

В отличие от мульти-процессного метода, многопоточный создаёт несколько однопоточных процессов.

Гибридный метод – это комбинация двух вышеупомянутых подходов. При таком подходе создается несколько процессов, и каждый процесс инициирует несколько потоков. Каждый из потоков обрабатывает одно соединение. Использование нескольких потоков в одном процессе приводит к меньшей нагрузке на системные ресурсы [5].

Почтовый сервер – это сервер, который обрабатывает и доставляет электронную почту по сети, обычно через Интернет. Почтовый сервер может

получать электронные письма с клиентских компьютеров и доставлять их на другие почтовые серверы. Почтовый сервер также может доставлять электронную почту на клиентские компьютеры. Клиентский компьютер - это, как правило, компьютер, на котором вы читаете свою электронную почту, например, компьютер дома или в офисе. Кроме того, в этих обстоятельствах в качестве клиентского компьютера можно рассматривать усовершенствованный мобильный телефон или смартфон.

На рисунке 1.4 иллюстрировано, каким образом осуществляется прием и передача электронного сообщения.

Рисунок 1.4 – «Движение» электронного сообщения

SMTP – это протокол прикладного уровня. Клиент, который хочет отправить почту, открывает TCP-соединение с SMTP-сервером, а затем отправляет почту через соединение. SMTP-сервер всегда находится в режиме прослушивания. Как только он прослушивает TCP-соединение от любого клиента, процесс SMTP инициирует соединение через этот порт (25). После успешного установления TCP-соединения клиентский процесс отправляет почту мгновенно.

IMAP и POP – два способа доступа к электронной почте. IMAP - это рекомендуемый метод, когда вам нужно проверить свою электронную почту с нескольких разных устройств, таких как телефон, ноутбук и планшет. IMAP позволяет вам получить доступ к электронной почте, где бы вы ни находились, с любого устройства. Когда вы читаете сообщение электронной почты с использованием IMAP, вы фактически не загружаете и не храните его на своем компьютере; вместо этого вы читаете это из почтовой службы. В результате вы можете проверить свою электронную почту с разных устройств в любой точке мира.

IMAP загружает сообщение только при нажатии на него, а вложения не загружаются автоматически. Таким образом, вы сможете проверять свои сообщения намного быстрее, чем POP.

POP работает, связываясь с вашей службой электронной почты и загружая все ваши новые сообщения с нее. Как только они загружаются на ПК, они удаляются из службы электронной почты. Это означает, что после загрузки электронной почты к ней можно получить доступ только с того же компьютера. Если вы попытаетесь получить доступ к своей электронной почте с другого устройства, ранее загруженные сообщения не будут вам

доступны. Отправленная почта хранится локально ПК, а не на почтовом сервере [4].

Вывод

В данной главе я дал определение понятию облачных вычислений, рассмотрел определение облачных вычислений, которое дал Национальный институт стандартов и технологий (NIST). Облачные вычисления – это модель, предоставляющая доступ к ресурсам по сети Интернет, которая поддерживает высокую доступность сервисов и описывается пятью основными характеристиками (essential characteristics), тремя сервисными моделями/моделями предоставления услуг (service models) и четырьмя моделями развертывания (deployment models).

Описал сервис Amazon EC2 и основные компоненты, такие как, инстансы, группа безопасности AWS, регионы и зоны доступности, Amazon Elastic Block Store. Подробно описал процесс подключения к инстансам EC2.

Также рассмотрел понятие WEB и почтовых серверов, описал их работу. Перечислил подходы WEB-сервера для обработки запросов – параллельный подход и подход основанный на отдельных процессах. Параллельный подход позволяет WEB-серверу обрабатывать несколько клиентских запросов одновременно, что может быть достигнуто с помощью следующих методов:

- мульти-процессорный метод;
- многопоточный метод;
- гибридный метод.

Основная задача протокола SMTP заключается в том, чтобы обеспечивать передачу электронных сообщений (почту). Для работы через протокол SMTP клиент создаёт TCP соединение с сервером через порт 25. Затем клиент и SMTP сервер обмениваются информацией пока соединение не будет закрыто или прервано.

2Анализ и выбор сервера

В данной главе я проанализирую рынок WEB-серверов, их возможности и выберу WEB-сервер; опишу основные меры защиты серверов.

2.1 Выбор WEB-сервера

Было время, когда WEB-сервер Apache обслуживал около 60 процентов, а иногда даже больше, WEB-сайтов в мире. Этот процент с тех пор упал ниже 30 и все еще падает довольно значительным темпом. Между тем, WEB-сервер Microsoft IIS занимает довольно устойчивую, слегка растущую долю рынка, достигнув сегодня 26 процентов. Второе место с Microsoft IIS делит конкурент по имени NGINX (произносится как «engine-x»), который в настоящее время так же обслуживает около 26 процентов всех сайтов, и этот показатель постоянно растет примерно на один процент каждый год.

На рисунке 2.1 представлен график, отражающий долю рынка WEB-серверов на рынке.

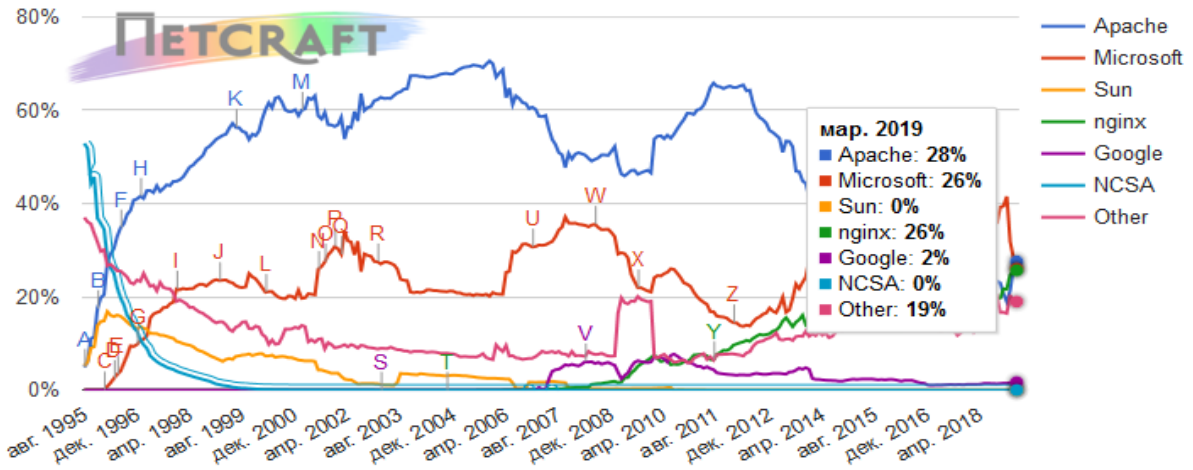

Рисунок 2.1 – Доля WEB-серверов на рынке

Apache. Apache – это компонент WEB-сервера популярного стека LAMP (Linux, Apache, MySQL, PHP). Несмотря на то, что в наши дни существует множество других компонентов WEB-стека (например, NodeJS, JS-среды для многофункциональных клиентов, различные облачные сервисы и т. Д.), LAMP по-прежнему остается очень популярным.

WEB-сервер Apache имеет богатый набор функций, которые можно включить, установив один из примерно 60 официальных модулей или один из множества других неофициальных модулей, которые также существуют.

За эти годы Apache разработал несколько методов обработки WEB-запросов для повышения его эффективности (главным образом, использования ОЗУ и задержки). В мире, где сайты должны обрабатывать все больше одновременных WEB-запросов, и где размеры обслуживаемых страниц становятся намного больше, необходимы были новые методологии.

Методология обработки запросов Apache может быть настроена одним из трех способов. Ниже я приведу три основных модуля мультипроцессинга (MPM):

- модель процесса – это оригинальный метод «pre-fork»; он плохо масштабируется со многими одновременными соединениями, так как он потребляет много оперативной памяти и может даже отказаться от соединений при высоких нагрузках.
- модель воркера – этот метод создает единый процесс управления, который отвечает за запуск дочерних процессов. Каждый дочерний процесс затем создает фиксированное количество потоков, а также поток слушателя. Поток слушателя прослушивает соединения и передает их потоку для обработки, когда они поступают. Хотя эта модель масштабируется намного лучше, чем метод «pre-fork», она все же может столкнуться с проблемами масштабирования для сайтов с очень большим трафиком из-за единого узкого места процесса управления.
- модель события – это похоже на рабочую модель, но она создает один поток слушателя, который прослушивает соединения и передает их в рабочий поток для обработки. Этот MPM намного эффективнее обрабатывает длительные соединения в одном потоке (обработка KeepAlive). Начиная с Apache 2.4, модель событий считалась стабильной и теперь также является настройкой по умолчанию, если операционная система может поддерживать ее.

Nginx. NGINX был создан в ответ на задачу C10K по обработке как минимум 10 000 одновременных клиентских подключений на одном сервере. NGINX использует асинхронную архитектуру, управляемую событиями, для обработки такого огромного количества соединений. Эта архитектура делает обработку высоких и колеблющихся нагрузок гораздо более предсказуемой с точки зрения использования ОЗУ, использования ЦП и задержки.

Основное различие между NGINX и Apache с точки зрения моделей событий заключается в том, что NGINX не устанавливает дополнительные рабочие процессы на соединение. В большинстве случаев рекомендуемая конфигурация NGINX - это запуск одного рабочего процесса на процессор, что максимизирует эффективность оборудования.

NGINX также имеет богатый набор функций и может выполнять различные роли сервера:

- обратный прокси-сервер для протоколов HTTP, HTTPS, SMTP, POP3 и IMAP;
- балансировщик нагрузки и HTTP-кеш;
- интерфейсный прокси для Apache и других WEB-серверов, сочетающий гибкость Apache с хорошей производительностью статического содержимого NGINX.

NGINX поддерживает обработчики FastCGI и SCGI для обслуживания сценариев динамического содержимого, таких как PHP и Python. Он

использует стек LEMP: разновидность LAMP с использованием фонетического написания NGINX (Linux, «En-juhn-ex», MySQL, PHP).

Сравнение Apache и Nginx. Apache на каждый запрос от клиента создает отдельный процесс (или поток, зависит от выбранного mpm модуля). Выглядит это следующим образом - клиент отправляет запрос, WEB-сервер создает отдельный процесс на этот запрос, отвечает клиенту и блокирует процесс до тех пор, пока клиент не закроет соединение. Процесс в любой ОС требует памяти и ресурсов, а когда процессов становиться неприлично много, обработка соединений неприлично замедляется, память кончается, CPU растет. Для мелких проектов такая реализация архитектуры обработки соединений не добавит головной боли, но для высоконагруженных проектов придется ставить очень мощное железо или искать альтернативные варианты.

Nginx состоит из master-процесса и нескольких дочерних процессов. Мастер процесс обычно один — он создает дочерние процессы (воркеры, загрузчик кеша и кеш менеджер), считывает конфигурацию и открывает порты. Воркеров обычно несколько, разработчики nginx советуют количество воркеров определять равным числу ядер машины. Эти дочерние процессы буду обслуживать все соединения с клиентами в неблокирующей манере. В nginx используется бесконечный цикл, который бежит по всем соединениями и отвечает на запросы клиентов. Когда соединение закрывается, оно удаляется из event loop. Это решение идеально подходит для проектов, которые обслуживающих более 10000 соединений одновременно. При этом, загрузка CPU и использование памяти обычно равномерны, без видимых пиков.

При отдачи статического контента NGINX приблизительно в 2,5 раза быстрее, чем Apache, основываясь на результатах тестов производительности, выполняющих до 1000 одновременных подключений. Другой тест с 512 одновременными подключениями показал, что NGINX работает примерно в два раза быстрее и потребляет немного меньше памяти (4%). Потребление памяти серверов можно увидеть на рисунке 2.2.

Рисунок 2.2 –Потребление памяти WEB-серверов Apache и Ngnix

Во время тестирования отдачи динамического контента, Apache и Nginx показывают примерно одинаковые результаты. Причиной этого является то, что почти все запросы обрабатываются в среде выполнения PHP, а не в основной части WEB-сервера. С точки зрения PHP (и, вероятно, других языков), производительность сервера динамических страниц практически равна при правильной настройке модуля Apache (PHP-FPM + FastCGI).

Оба проекта имеют отличную репутацию в области безопасности для своей базы кода на C. Однако кодовая база NGINX значительно меньше на несколько порядков, что, безусловно, является большим плюсом с дальновидной точки зрения безопасности. Доступны отчеты об уязвимостях для Apache 2.2 и 2.4. NGINX также имеет список последних рекомендаций по безопасности. Так же, в отличие от Apache, модули Nginx не могут быть динамически загружены на лету и требуют сборки. Это сложнее, но считается безопаснее [5].

2.2 Меры защиты серверов

Исследования показывают, что каждый день взламывается 37 000 WEB-сайтов, что соответствует 13,5 миллионам WEB-сайтов в год. Злоумышленники используют широкий спектр методов для распространения своих вредоносных программ от использования уязвимостей программного обеспечения до заражения рекламных сетей. Обычного межсетевого экрана недостаточно для безопасности WEB-сервера. Поэтому необходимо развернуть серию защит, одну за другой .

В первую очередь необходимо позаботиться о безопасном подключении к WEB-серверу. Использование SSH не решает проблему нелегитимного доступа к серверу. Ведь, если, разрешена аутентификация на основе пароля, злоумышленники могут добраться до данных сервера. Именно по этой причине рекомендуется использовать ssh-ключи для авторизаций. В основе технологии – пара криптографических ключей, которые используют для проверки подлинности в качестве альтернативы аутентификации с помощью пароля. Система входа использует закрытый и открытый ключи, которые создают до аутентификации. Закрытый ключ хранится в тайне надежным пользователем, в то время как открытый ключ может раздаваться с любого сервера SHH, к которому нужно подключиться [7].

Избавиться от подавляющего большинства сетевых атак можно отключив сетевые службы, закрыв все порты, за исключением, только тех, которые действительно нужны - это самый фундаментальный принцип в сетевой безопасности. Запретить всё и разрешите только то, что нужно [8].

Брандмауэр WEB-приложений – неотъемлемая деталь механизма защиты WEB-серверов. Он фильтрует, отслеживает и блокирует HTTP-трафик в WEB-приложение и из него. WAF отличается от обычного брандмауэра тем, что WAF может фильтровать содержимое определенных WEB-приложений, в то время как обычные брандмауэры служат в качестве шлюза безопасности между серверами. Проверяя HTTP-трафик, он может предотвратить атаки, связанные с уязвимостями безопасности WEB-приложений, такими как внедрение SQL-инъекций, межсайтовый скриптинг (XSS), внедрение файлов и неправильная конфигурация безопасности.

Все WEB-серверы оптимизированы для обеспечения функциональности в конфигурации по умолчанию. Это представляет злоумышленнику большую область атаки. Итак, обеспечение безопасности конфигурации сервера

является первым шагом в безопасности WEB-службы. В конфигурации по умолчанию большинство WEB-серверов отображают большое количество информации о версии программного обеспечения, списке файлов в каталоге, списке модулей WEB-сервера и т. д. Если эти настройки явно не отключены, они предоставляют большое количество информации для атакующего. Исходя из совокупности всех факторов, необходимо предотвратить отображение этой информации. Также нужно отредактировать конфигурационный файл для защиты от распространенных атак, как Slowloris, Syn flood.

Рекомендуется установка WEB-сервера только с необходимыми модулями и функциями, чтобы злоумышленники не использовали уязвимости неиспользуемых модулей; удаление файлов WEB-страницы по умолчанию, сценарии cgi по умолчанию и каталог руководства WEB-сервера, поскольку они содержат информацию о WEB-сервере.

Следует ограничить доступные методы HTTP. Подавляющее большинство WEB-приложений используют запрос «GET», запрос «POST» или «HEAD» для отображения WEB-страниц. По умолчанию WEB-серверы позволяют использовать другие типы запросов, такие как «DELETE», «TRACE», «TRACK» и т.д., которые могут использоваться злоумышленниками для получения системной информации. По этой причине их можно отключить.

Если сайт работает с приватными данными пользователей, такими как номера кредитных карт, пароли от других сервисов, или же предоставляет доступ к другой важной информации, которую может заполучить злоумышленник, необходимо позаботься о шифровании. Рекомендуется использовать SSL/TLS для шифрования данных, чтобы защититься от атак типа man-in-the-middle [10].

Использование CDN, а именно пограничных серверов, для сокрытия настоящего ip-адреса сервера, может быть хорошей мерой защиты. В настоящее время многие CDN предоставляют услуги по защите от DDoS-атак абсолютно бесплатно, что так же будет нелишним для безопасности WEB-сервера.

Использование сетевых сканеров безопасности для выявления небезопасных конфигураций, отсутствия необходимых обновлений и наличия уязвимостей серверов может стать хорошей точкой в защите-WEB сервера.

Для защиты почтового сервера, в первую очередь необходимо добавить и настроить SPF ,DKIM записи и механизм DMARC, которые могут помочь предотвратить подделку электронной почты и гарантировать, что легитимные электронные письма будут доставляться в папку «Входящие» получателя, а не в папку «Спам» [6].

Так же как и в случае с WEB-сервером, рекомендуется использование шифрования для защиты сообщений электронной почты – даже если они будут перехвачены, их содержимое будет невозможно прочесть.

При борьбе со спамом можно воспользоваться следующими правилами:

- отклонить электронную почту, если у клиента SMTP нет записи PTR;

- включить ограничения HELO / EHLO Hostname в Postfix;
- отклонить электронную почту, если имя хоста клиента SMTP не имеет действительной записи A;
- отклонить электронное письмо, если в MAIL FROM домена нет ни Mx записи, ни A записи;
- запуск Greylisting в Postfix;
- использование общедоступных антиспамовых черных списков
- блокировать спам в электронной почте, проверяя заголовок и текст сообщения.

Вывод

В данной главе я рассмотрел и сравнил WEB-сервера Apache и Nginx. Я выбрал Nginx в качестве WEB-сервера потому, что на сегодняшний день он набирает популярность и оттесняет Apache с лидирующей позиции в рейтинге используемых серверов по всему миру, с точки зрения производительности, если речь идет об обработке статического контента, оптимальности использования ресурсов Nginx так же превосходит Apache. Скорость обработки динамического контента у этих WEB-серверов одинакова в том случае, когда Apache настроена правильно, то есть это лишние исправления конфигурационного файла, которых при настройке Nginx можно избежать.

Я не сравнивал почтовые серверы, так как сравнил Apache и Nginx, по той причине, что все доступные бесплатные почтовые серверы имеют идентичную архитектуру и функционал. По этой причине мой выбор пал в сторону iRedMail, который автоматически устанавливает и настраивает все необходимые компоненты почтового сервера.

Также в данной главе я описал меры защиты WEB и почтовых серверов. Защита WEB серверов, в основном, заключается в предотвращении утечки информации, которая может послужить «точкой входа» для злоумышленника. Защита почтовых серверов, в основном, заключается в фильтрации спама, обеспечении успешной и безопасной доставки легитимных исходящих писем, предотвращении использования доменного имени злоумышленниками для отправки спама, использовании антивирусов для сканирования входящей почты на содержание вредоносных вложенных файлов.

3 Практическая часть

В данной главе будет рассмотрен процесс развертывания и настройки виртуальных серверов Amazon EC2, настройка и обеспечение безопасности WEB и почтовых серверов.

3.1 Развертывание и настройка инстанса Amazon EC2

Первым делом при создании инстанса Amazon EC2 нужно выбрать операционную систему и тип инстанса. Я выбираю Ubuntu server 18,04 в качестве ОС, и t2 micro в качестве инстанса (рисунок 3.1):

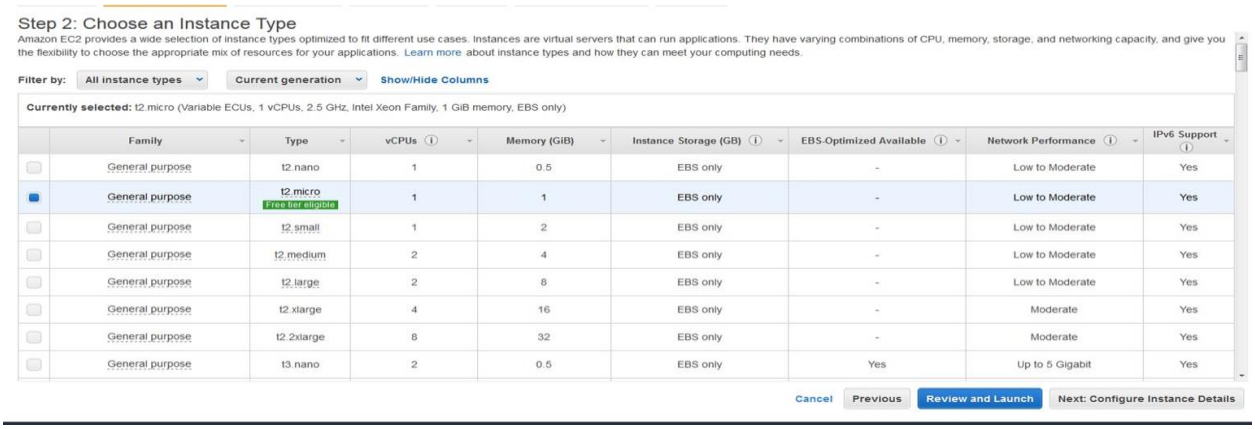

Рисунок 3.1 – Доступные типы инстансов Amazon EC2

Далее необходимо настроить группы безопасности AWS таким образом, чтобы входящий трафик подходил для WEB и почтовых серверов (рисунок 3.2).

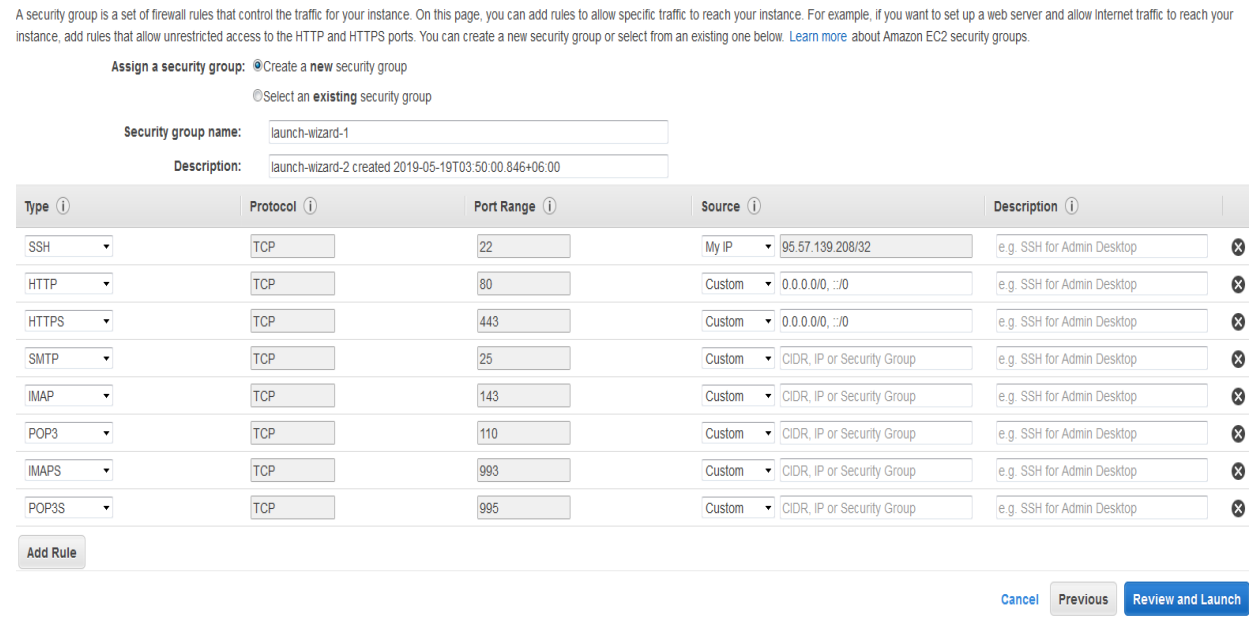

Рисунок 3.2 – Настройка группы безопасности AWS

На следующем шаге необходимо создать пару ключей (открытого и закрытого), скачать себе закрытый ключ, который будет постоянно использоваться для подключения к инстансу по SSH (рисунок 3.3).

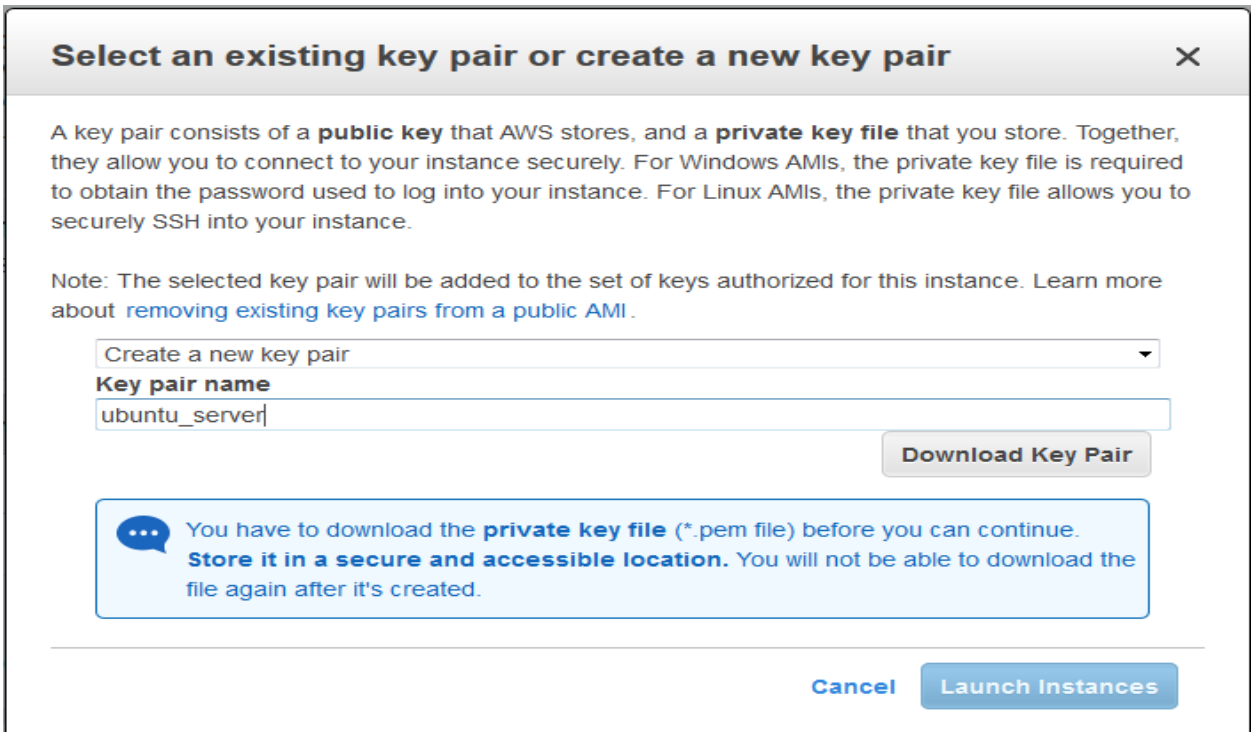

Рисунок 3.3 – Создание пары ключей

3.2 Настройка и обеспечение безопасности WEB-сервера Ngnix

Обеспечение безопасности WEB-сервера начинается с этапа установки Nginx, так как мы можем собрать Ngnix из исходного кода, исключив ненужные и потенциально опасные модули, и включив полезные модули, как WAF(рисунок 3.4).

```
bayanaman@ip-172-31-33-120:~/nginx-1.15.12$ ./configure \
>       --conf-path=/etc/nginx/nginx.conf \
>       --add-module=../naxsi-0.56/naxsi_src/ \
>       --error-log-path=/var/log/nginx/error.log \
>       --http-client-body-temp-path=/var/lib/nginx/body \
>       --http-fastcgi-temp-path=/var/lib/nginx/fastcgi \
>       --http-log-path=/var/log/nginx/access.log \
>       --http-proxy-temp-path=/var/lib/nginx/proxy \
>       --lock-path=/var/lock/nginx.lock \
>       --pid-path=/var/run/nginx.pid \
>       --user=www-data \
>       --group=www-data \
>       --with-http_ssl_module \
>       --without-mail_pop3_module \
>       --without-mail_smtp_module \
>       --without-mail_imap_module \
>       --without-http_uwsgi_module \
>       --without-http_scgi_module \
>       --prefix=/usr
checking for OS
```

Рисунок 3.4 – Сборка Nginx из исходного кода

После сборки устанавливаем правила блокировки для защитного экрана уровня приложения naxsi (рисунок 3.5), и включаем эти правила в конфигурационном файле nginx.conf (рисунок 3.6).

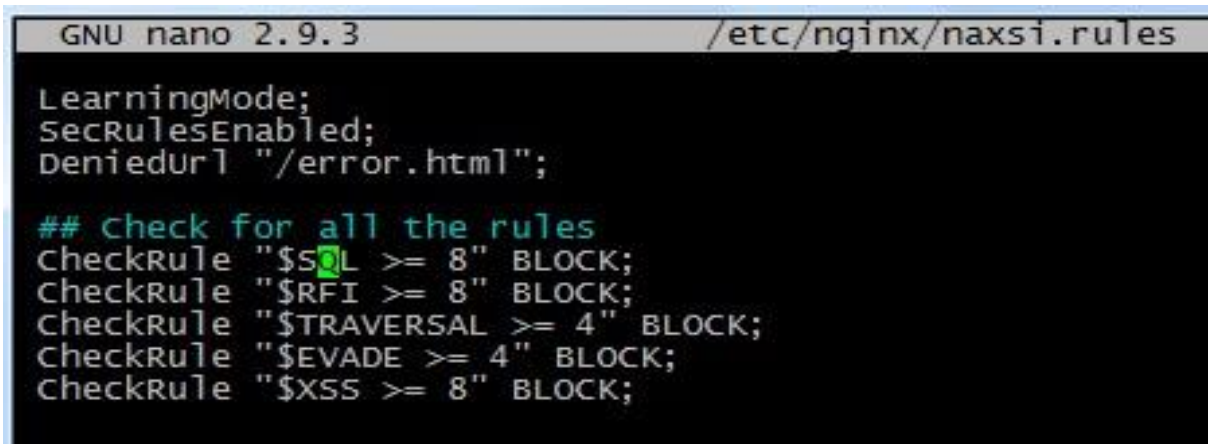

Рисунок 3.5 – Список правил блокировки naxsi

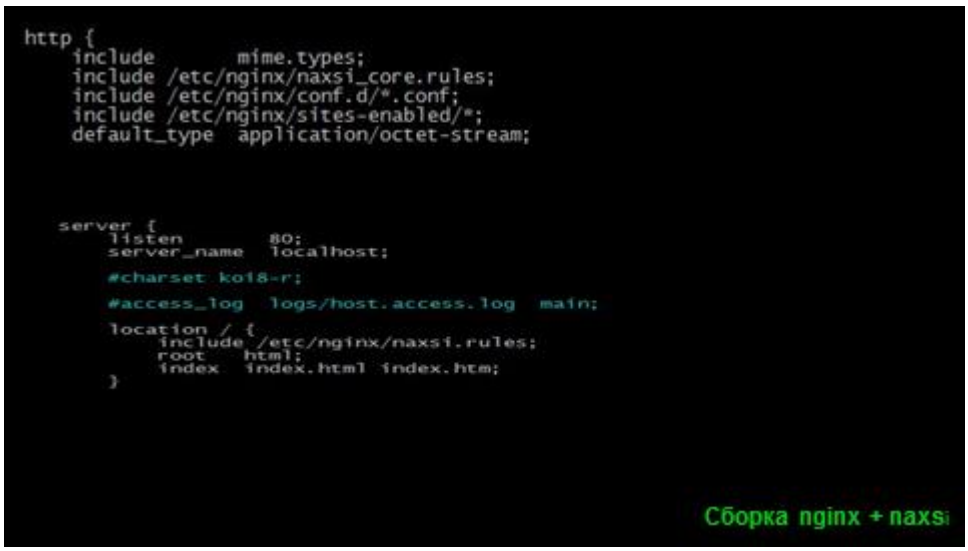

Рисунок 3.6 – Включаем возможности naxsi

Отправив серверу «подозрительный» запрос, тестируем работоспособность naxsi (рисунок 3.7). Логи naxsi можно увидеть на рисунке 3.8.

Рисунок 3.7 – результат работы naxsi

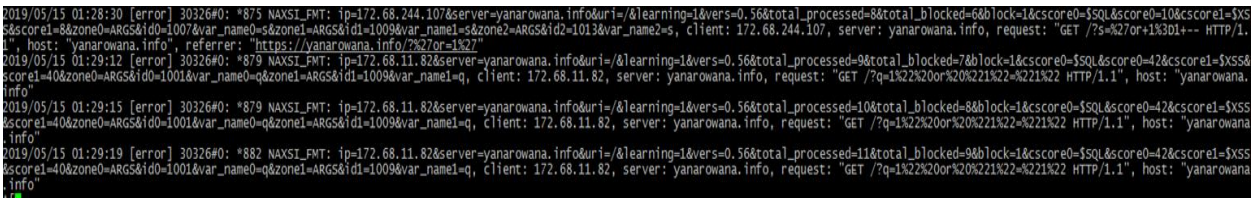

Рисунок 3.8 – Логи naxsi

Для использования безопасного подключения нам нужен SSL/TLS сертификат. Компания Cloudflare позволяет создать бесплатный сертификат TLS подписанный Cloudflare, для установки на сервер. Всё что от нас требуется это установка открытого и закрытого ключа на сервере (рисунок 3.9, рисунок 3.10), включить опцию, позволяющую использовать SSL и указать путь к ключам в файле конфигурации WEB-сервера (рисунок 3.11).

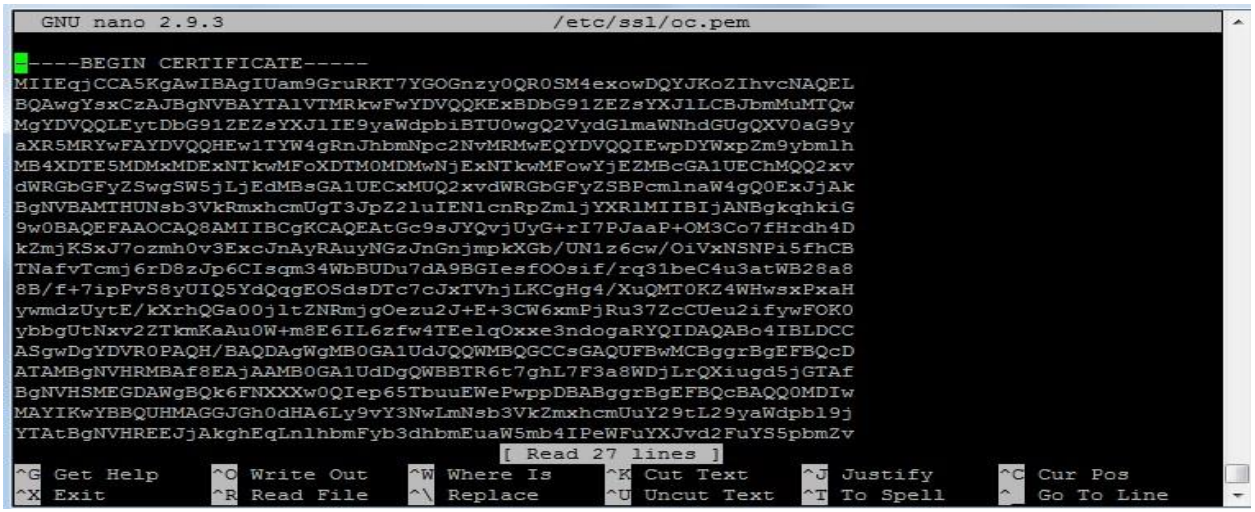

Рисунок 3.9 – Файл открытого ключа

Рисунок 3.10 – Файл закрытого ключа

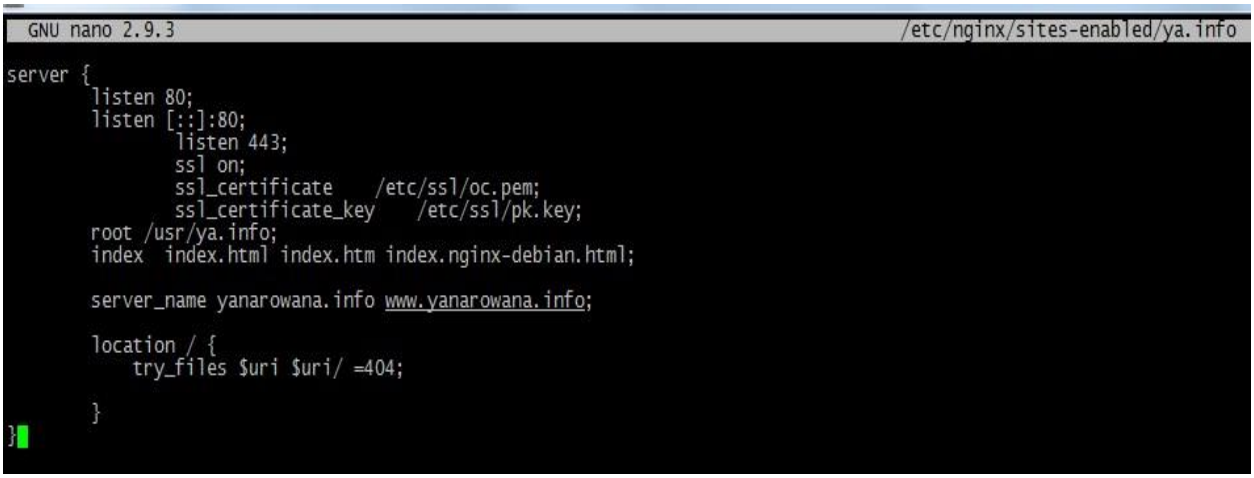

Рисунок 3.11 – Настройка конфигурационного файла для использования SSL

При посещении сайта можно удостовериться, что подключению осуществляется с использованием протокола HTTPS. Параметры сертификата представлены на рисунке 3.12.

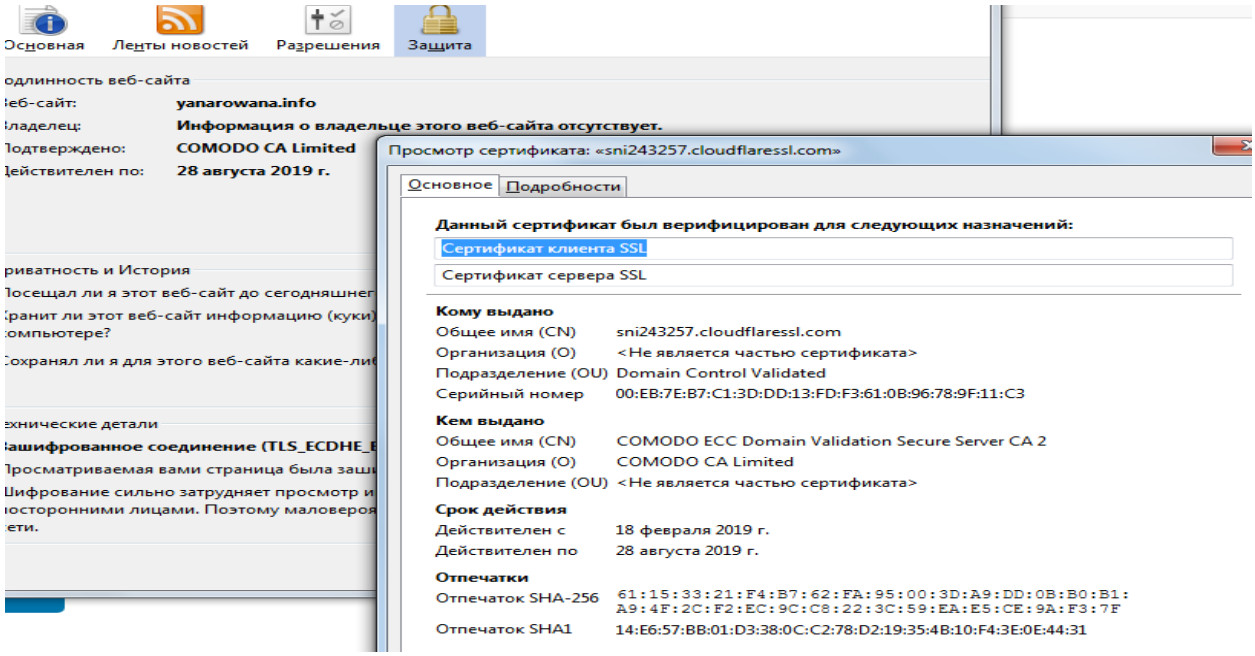

Рисунок 3.12 – Подробности о сертификате

Для защиты от атак типа Session Attack, в данном случае SlowLoris, можно закрыть соединение по истечению 5 секунд, если клиент не отвечает (рисунок 3.13). Обычно боты просто перебирают диапазоны IP-адресов в поисках открытых 80 портов и посылают запрос HEAD для получения информации о WEB-сервере. Можно легко предотвратить такой скан, запретив обращение к серверу по IP-адресу (рисунок 3.14). Некоторые боты используют различные методы обращения к серверу для попытки

определения его типа и/или проникновения, по этой причине все методы, кроме GET, HEAD и POST, можно безболезненно отключить (рисунок 3.14).

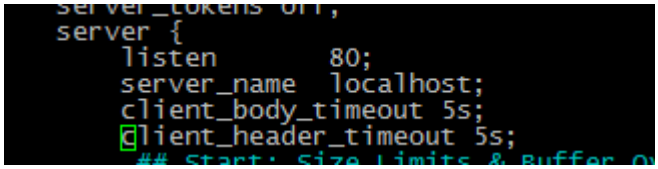

Рисунок 3.13 – Настройка снижения тайм-аута на ожидание ответного пакета

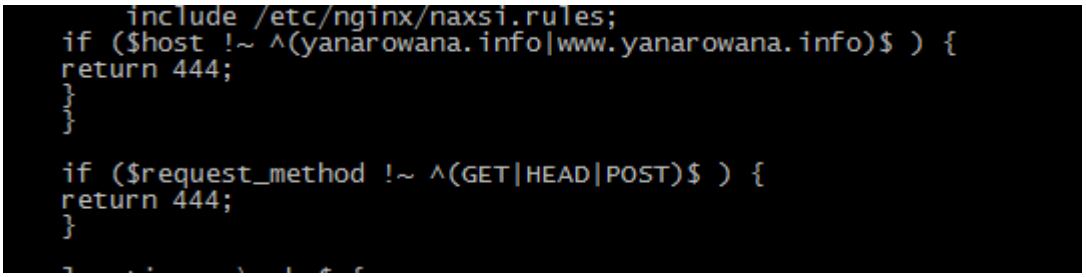

Рисунок 3.14 – Настройка запрета обращения к серверу по ip-адресу и отключение необязательных методов

CDN Cloudflare помогает поглощать поток трафика, связанный с атаками DDoS. В дополнение к этой встроенной защите от DDoS, Cloudflare обеспечивает дополнительную защиту «Under Attack Mode» (рисунок 3.15). Это уровень безопасности, который вы включаете, когда ваш сайт подвергается активной атаке. Когда этот режим включен, он добавляет дополнительные средства защиты, чтобы предотвратить попадание потенциально опасного HTTP-трафика на сайт. Результат работы этого режима представлен на рисунке 3.16.

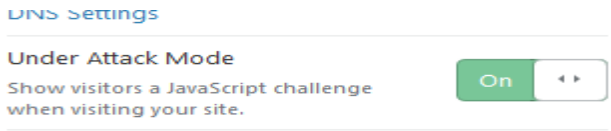

Рисунок 3.15 – Режим Under Attack Mode

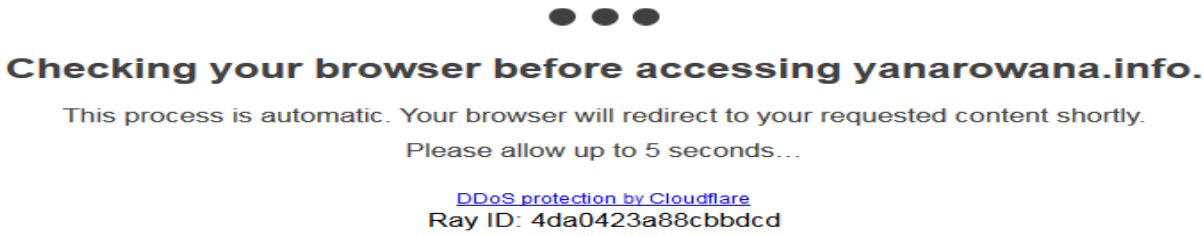

Рисунок 3.16 – Результат работы режима Under Attack Mode

Настройка стека LEMP и CMS Wordpress представлена на рисунке 3.17:

```
server {
        listen 80;
        listen [::]:80;
                listen 443 ssl;
                ssl_certificate     /etc/ssl/oc.pem;
                ssl_certificate_key     /etc/ssl/pk.key;
        root /usr/ya.info;
        index index.php index.html index.htm index.nginx-debian.html;

        server_name yanarowana.info www.yanarowana.info;

        location / {
            #try_files $uri $uri/ =404;
            try_files $uri $uri/ /index.php$is_args$args;
            include /etc/nginx/naxsi.rules;
        }
        location ~ \.php$ {
                include fastcgi.conf;
                fastcgi_pass unix:/var/run/php/php7.2-fpm.sock;
        }

        location ~ /\.ht {
                deny all;
        }
       location = /favicon.ico { log_not_found off; access_log off; }
       location = /robots.txt { log_not_found off; access_log off; allow all; }
       location ~* \.(css|gif|ico|jpeg|jpg|js|png)$ {
        expires max;
        log_not_found off;
       }
}
```

Рисунок 3.17 – Настройка LEMP и Wordpress

В Wordpress CMS необходимо установить плагин web application firewall Wordfence. Список правил Wordfence, настройки защиты от перебора, блокировки от ботов представлены на рисунках 3.18, 3.19, 3.20 соответственно.

	Category	Description
●	whitelist	Whitelisted URL
●	lfi	Slider Revolution: Local File Inclusion
●	sqli	SQL Injection
●	xss	XSS: Cross Site Scripting
●	file_upload	Malicious File Upload
●	lfi	Directory Traversal
●	lfi	LFI: Local File Inclusion
●	xxe	XXE: External Entity Expansion
●	xss	dzs-videogallery 8.80 XSS HTML injection in inline JavaScript
●	sqli	Simple Ads Manager <= 2.9.4.116 - SQL Injection
●	rfi	Gwolle Guestbook <= 1.5.3 - Remote File Inclusion
●	priv-esc	User Roles Manager Privilege Escalation <= 4.24
●	auth-bypass	WordPress Core <= 4.5.0 - Authentication Bypass

Рисунок 3.18 – Список правил Wordfence

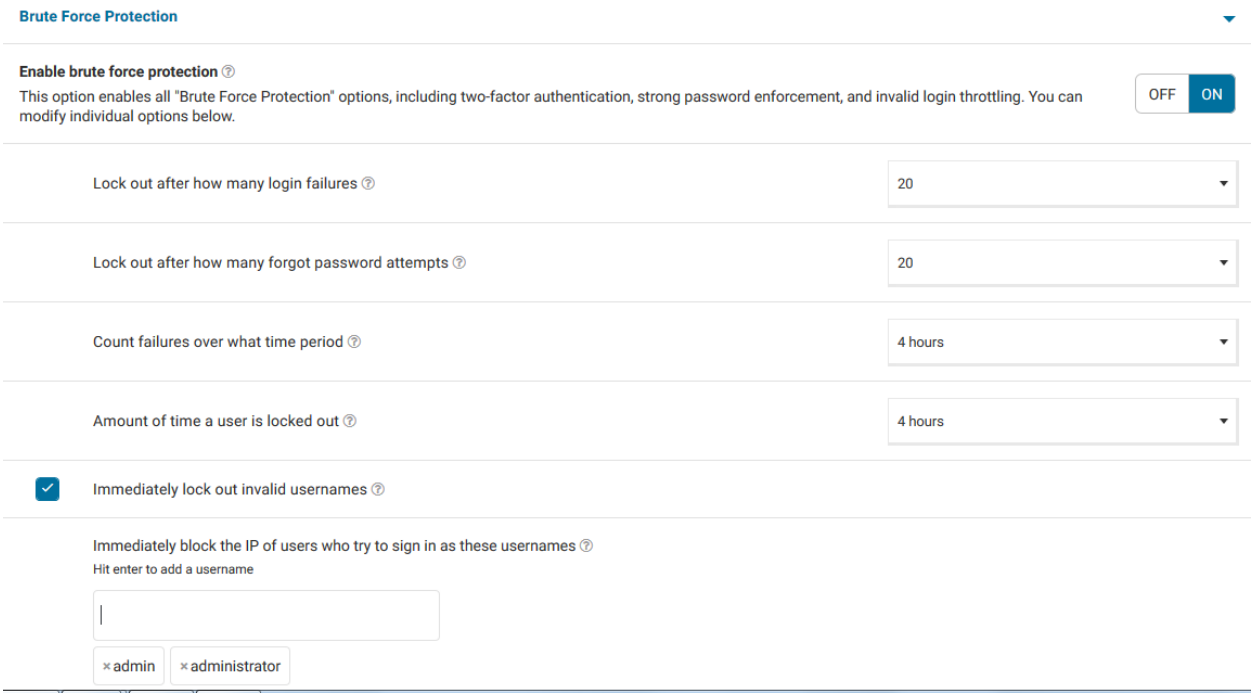

Рисунок 3.19 – Параметры защиты от перебора (brute force)

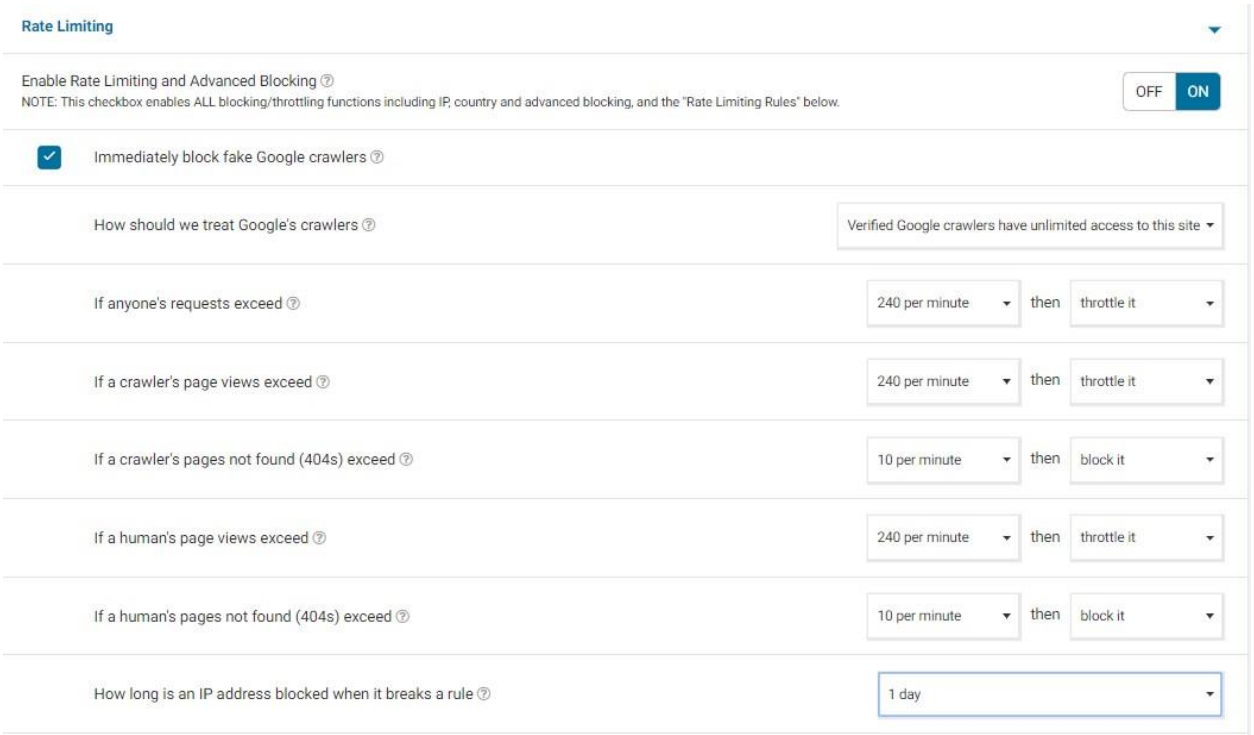

Рисунок 3.20 – Параметры блокировки ботов

Теперь после 20 попыток ввода неверного логина и пароля в течение 4 часов ip-адрес пользователя будет заблокирован на 4 часа, и пользователь не будет иметь возможность войти на сайт. Также если пользователь попытается войти в админ-панель используя логин «admin» или «administrator» он будет заблокирован немедленно (рисунок 3.21, рисунок 3.22).

Рисунок 3.21 – Попытка входа в систему, используя логин «admin»

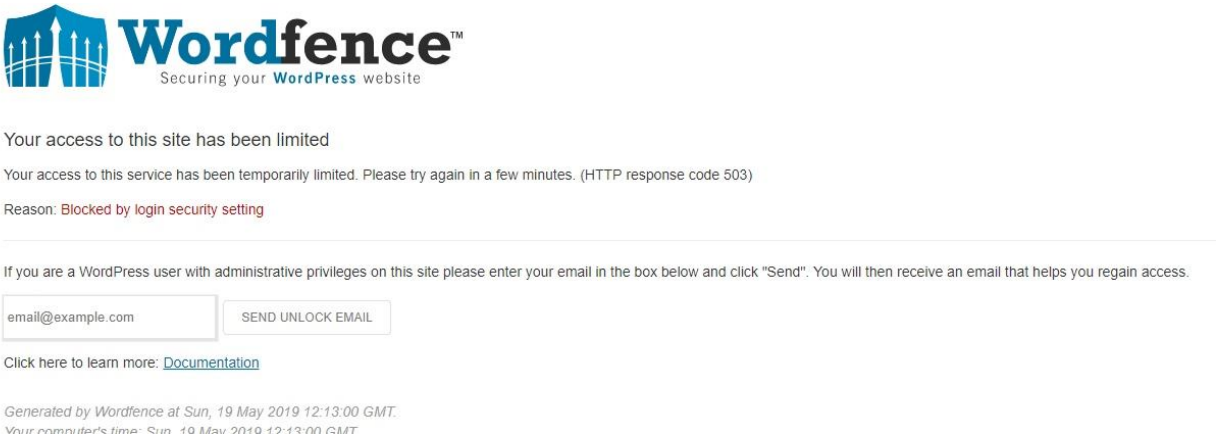

Рисунок 3.22 – Страница блокировки wordfence

Можем обезопасить наш сайт от brute force атак изменив стандартный url авторизации Wordpress wp-admin и wp-login.php на какой-нибудь другой, используя плагин Wps-Hide-Login. Изменяем стандартный url и сохраняем настройки (рисунок 3.23). Теперь при обращении по стандартному url /wp-admin или /wp-login.php мы получаем ошибку 404 (рисунок 3.24).

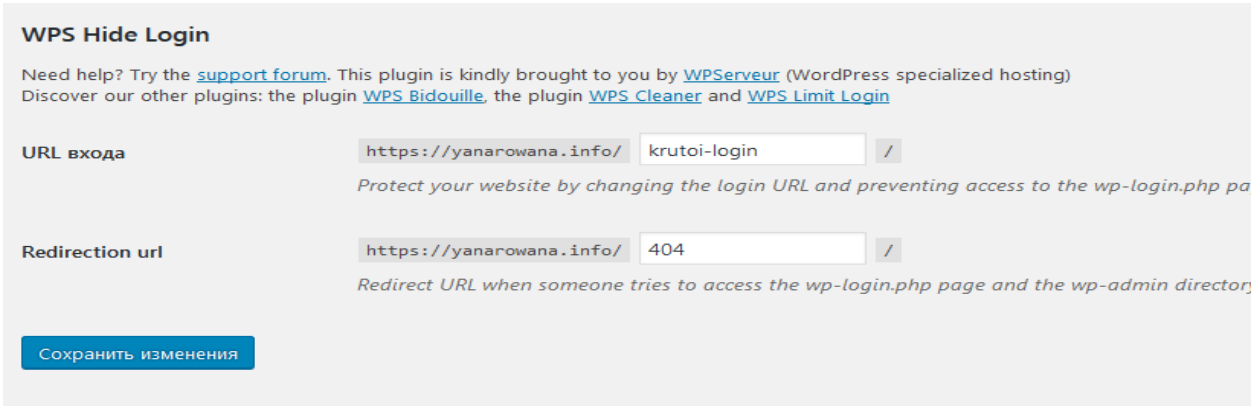

Рисунок 3.23 – Изменение стандартного URL страницы авторизаций

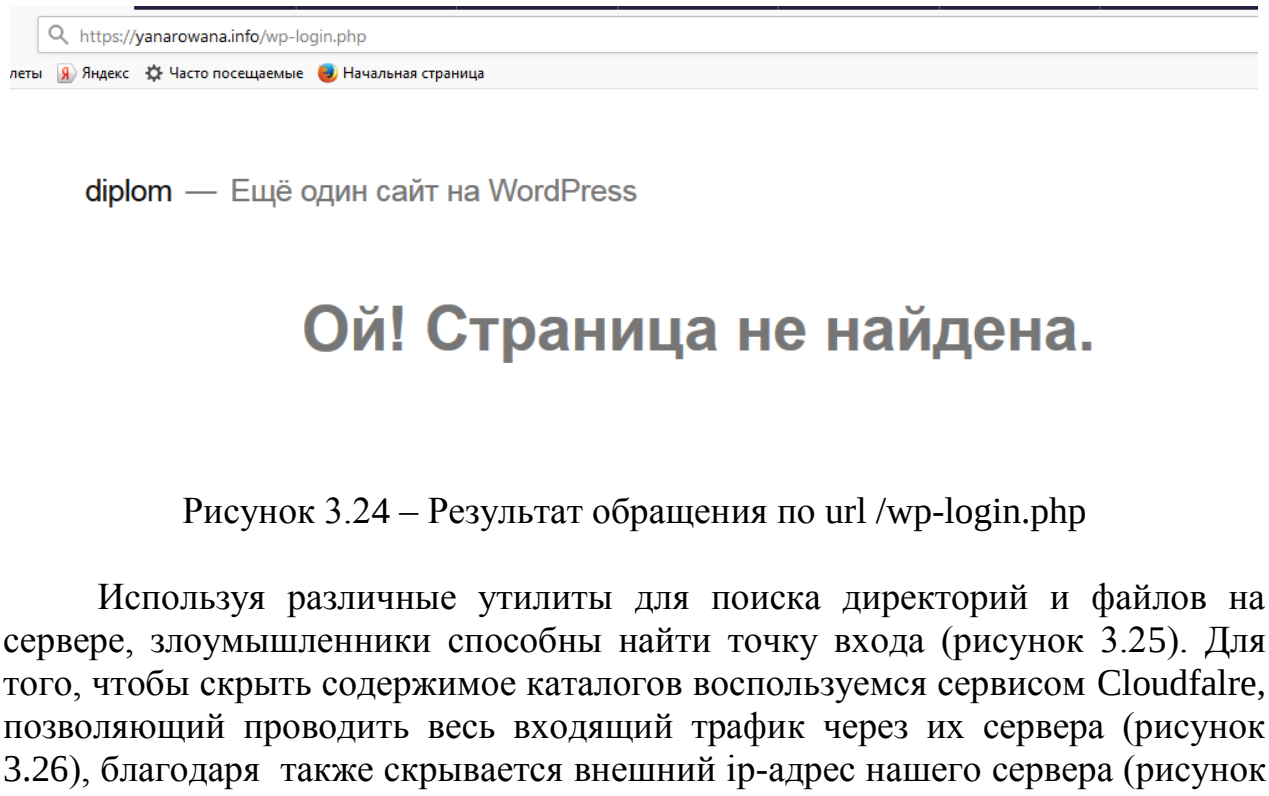

Рисунок 3.24 – Результат обращения по url /wp-login.php

Используя различные утилиты для поиска директорий и файлов на сервере, злоумышленники способны найти точку входа (рисунок 3.25). Для того, чтобы скрыть содержимое каталогов воспользуемся сервисом Cloudfalre, позволяющий проводить весь входящий трафик через их сервера (рисунок 3.26), благодаря также скрывается внешний ip-адрес нашего сервера (рисунок 3.27). Повторно просканировав наш сервер, можно обнаружить, что реальные директории и файлы были скрыты серверами cloudfalre (рисунок 3.28).

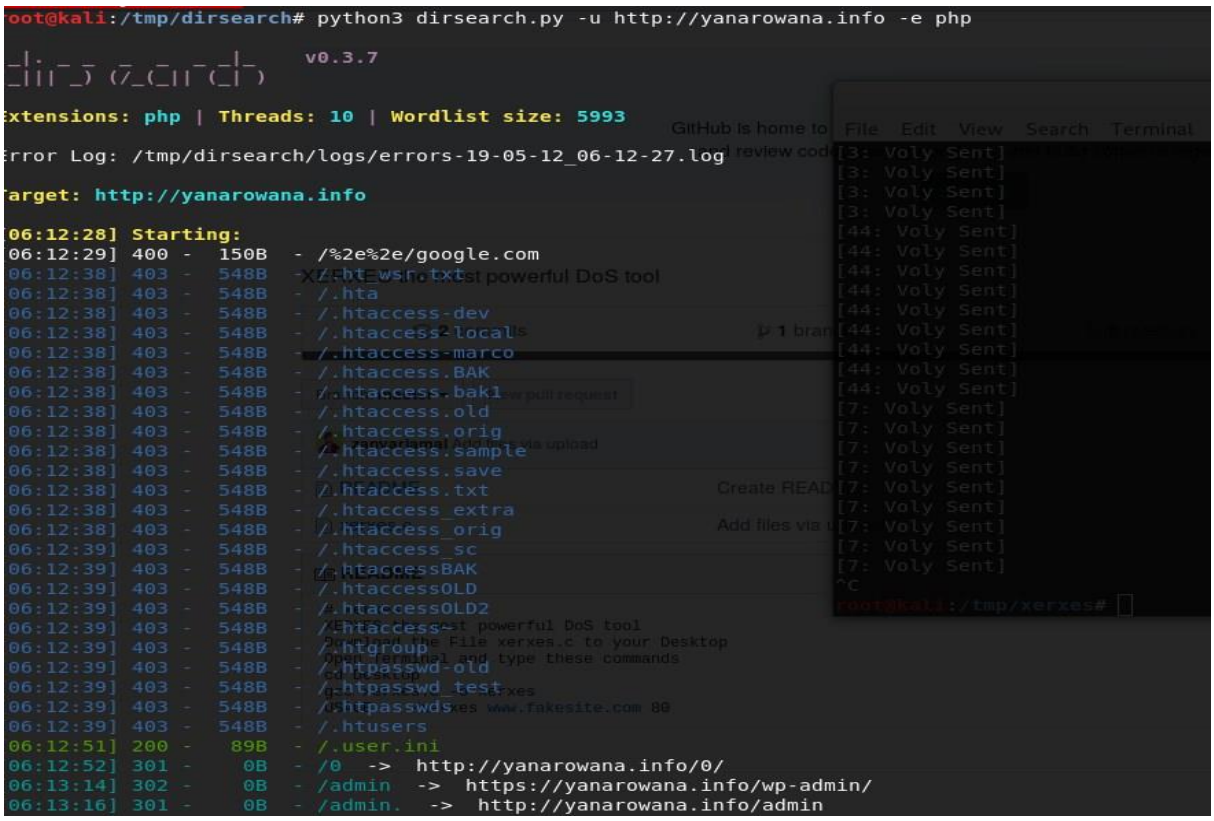

Рисунок 3.25 – Результат сканирования директорий и файлов

Type	Name	Value	TTL	Status	
A	mail	points to 46.19.41.22	Automatic		✕
A	yanarowana.info	points to 3.18.126.123	Automatic		✕
CNAME	ftp	is an alias of yanarowana.info	Automatic		✕
CNAME	www	is an alias of yanarowana.info	Automatic		✕
MX	yanarowana.info	mail handled by mail.yanarowana.info (10)	Automatic		✕
PTR	46.19.41.22	points to yanarowana.info	Automatic		✕
TXT	_dmarc	v=DMARC1; p=none; pct=100; rua=mailto:yan...	Automatic		✕
TXT	default._domainkey	v=DKIM1; h=sha256; k=rsa; p=MIIBIjANBgkqh...	Automatic		✕
TXT	yanarowana.info	v=spf1 +a +mx ip4:18.223.116.73 -all	Automatic		✕

Рисунок 3.26 – Настройка cloudlfare dns-proxy

```
C:\Users\никенке>nslookup yanarowana.info
Сервер:  UnKnown
Address:  192.168.1.1

Не заслуживающий доверия ответ:
Имя:    yanarowana.info
Addresses:  104.24.127.154
          104.24.126.154
```

Рисунок 3.27 – Результат работы nslookup

```
root@kali:/tmp/dirsearch# python3 dirsearch.py -u http://yanarowana.info -e php

 _|. _ _  _  _  _ _|_    v0.3.7
(_||| _) (/_(_|| (_| )

Extensions: php | Threads: 10 | Wordlist size: 5993

Error Log: /tmp/dirsearch/logs/errors-19-05-12_06-18-59.log

Target: http://yanarowana.info

[06:18:59] Starting:
[06:19:00] 301 -    0B  - /php  ->  https://yanarowana.info/php
[06:19:00] 400 -  171B  - /%2e%2e/google.com
[06:19:17] 301 -    0B  - /adminphp  ->  https://yanarowana.info/adminphp
CTRL+C detected: Pausing threads, please wait...
[e]xit / [c]ontinue: ^[^[
```

Рисунок 3.28 – Результат повторного сканирования директорий и файлов

Для того чтобы усложнить задачу для потенциального злоумышленника, пытающегося атаковать наш сайт с помощью sql-инъекций , рекомендуется изменить стандартный префикс таблиц Wordpress. Для этого в панели управления базами данных phpmyadmin экспортируем копию базы данных “wordpress” в формате .sql (рисунок 3.29), изменяем префикс “wp” на любой другой (рисунок 3.30), сохраняем результат и загружаем на сервер,

предварительно удалив струю базу данных (рисунок 3.31), изменяем префикс в конфигурационном файле wp-config.php (рисунок 3.32). Убедиться в том, что префикс изменился, можно сделав запрос выборки прямо на сервере (рисунок 3.33).

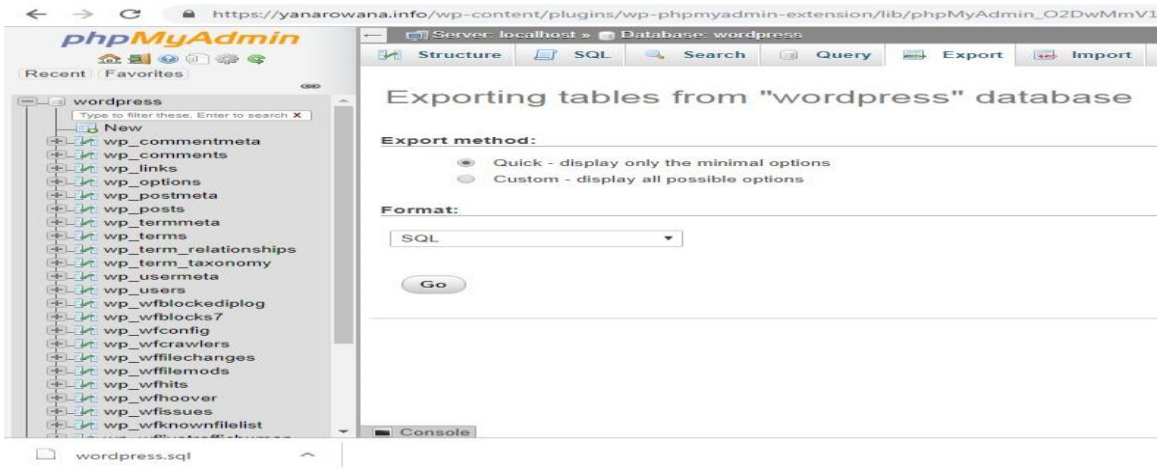

Рисунок 3.29 – Экспорт базы данных wordpress

```
SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET AUTOCOMMIT = 0;
START TRANSACTION;
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8mb4 */;

--
-- Database: `wordpress`
--

-- --------------------------------------------------------

--
-- Table structure for table `wp_commentmeta`
--

CREATE TABLE `krutoi_commentmeta` (
  `meta_id` bigint(20) UNSIGNED NOT NULL,
  `comment_id` bigint(20) UNSIGNED NOT NULL DEFAULT '0',
  `meta_key` varchar(255) COLLATE utf8mb4_unicode_520_ci DEFAULT NULL,
  `meta_value` longtext COLLATE utf8mb4_unicode_520_ci
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_520_ci;

-- --------------------------------------------------------
```

Рисунок 3.30 – Редактирование скрипта sql

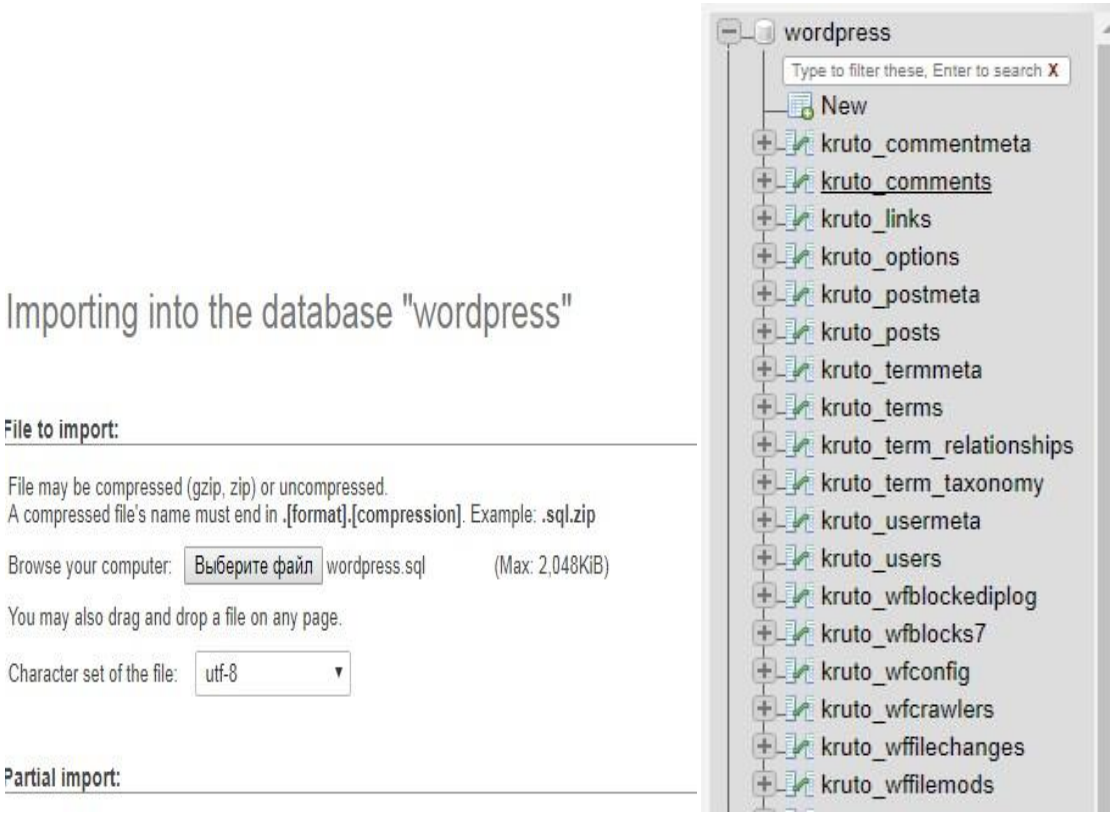

Рисунок 3.31 – Импорт отредактированной базы

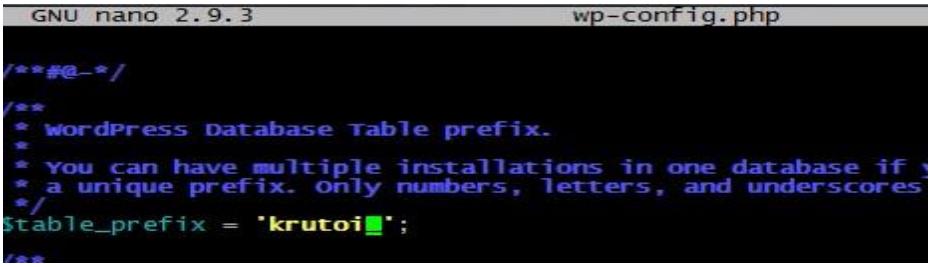

Рисунок 3.32 – Изменение префикса в конфигурационном файле wordpress

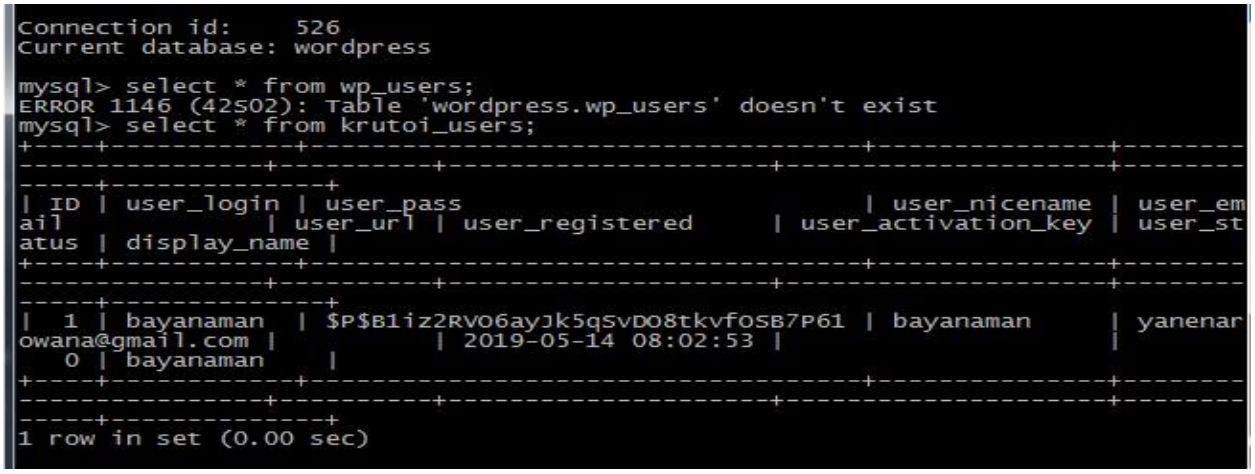

Рисунок 3.33 – Результат запроса выборки на сервере

Используя сканер уязвимостей, просканируем сайт на наличие уязвимостей (рисунок 3.34).

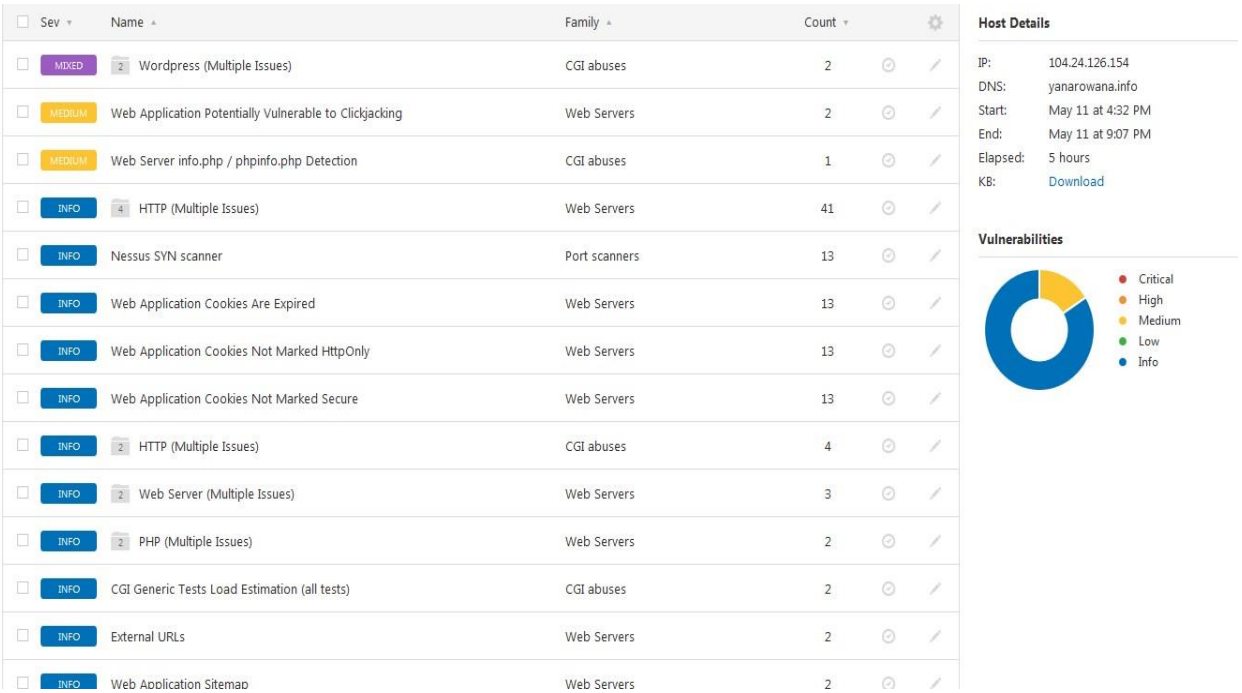

Рисунок 3.34 – Результат сканирования Nessus

Сканер нашел уязвимость user enumeration, благодаря которой злоумышленник способен узнать имена пользователей wordpress и использовать их для попытки взлома страницы авторизаций. Результат сканирования пользователей представлен на рисунке 3.35. Чтобы устранить эту «брешь», устанавливаем и активируем плагин Stop User Enumeration (рисунок 3.36). После повторного сканирования попытки найти имена пользователей не увенчались успехом (рисунок 3.37).

```
[i] User(s) Identified:

[+] bayanaman
 | Detected By: Author Posts - Author Pattern (Passive Detection)
 | Confirmed By:
 |  Rss Generator (Passive Detection)
 |  Wp Json Api (Aggressive Detection)
 |   - https://yanarowana.info/wp-json/wp/v2/users/?per_page=100&page=1
 |  Rss Generator (Aggressive Detection)
 |  Author Id Brute Forcing - Author Pattern (Aggressive Detection)
```

Рисунок 3.35 – Результат сканирования имен пользователей

Рисунок 3.36 – Установка плагина stop user enumeration

Рисунок 3.37 –Результат повторного сканирования имен пользователей

Следующая уязвимость, которую обнаружил сканер называется Clickjacking (рисунок 3.38). Clickjacking - это атака, которая вынуждает пользователя кликнуть на элемент WEB-страницы, который невидим или скрыт под иным элементом. Это может привести к тому, что пользователи будут невольно загружать вредоносные программы, посещать вредоносные WEB-страницы, предоставлять учетные данные или конфиденциальную информацию, переводить деньги или приобретать продукты через Интернет.

Для защиты от этой атаки используем Заголовок X-Frame-Options. Заголовок ответа X-Frame-Options передается как часть HTTP-ответа WEB-страницы, указывая, разрешено ли браузеру отображать страницу внутри тега <FRAME> или <IFRAME>. Я буду использовать значение SAMEORIGIN, позволяющее отображать текущую страницу в рамке на другой странице, но только в пределах текущего домена. Для того, чтобы включить эту опцию изменяем конфигурационный файл ngnix.conf (рисунок 3.39). На рисунках 3.40 и 3.41 представлены заголовки ответа до и после применения изменении.

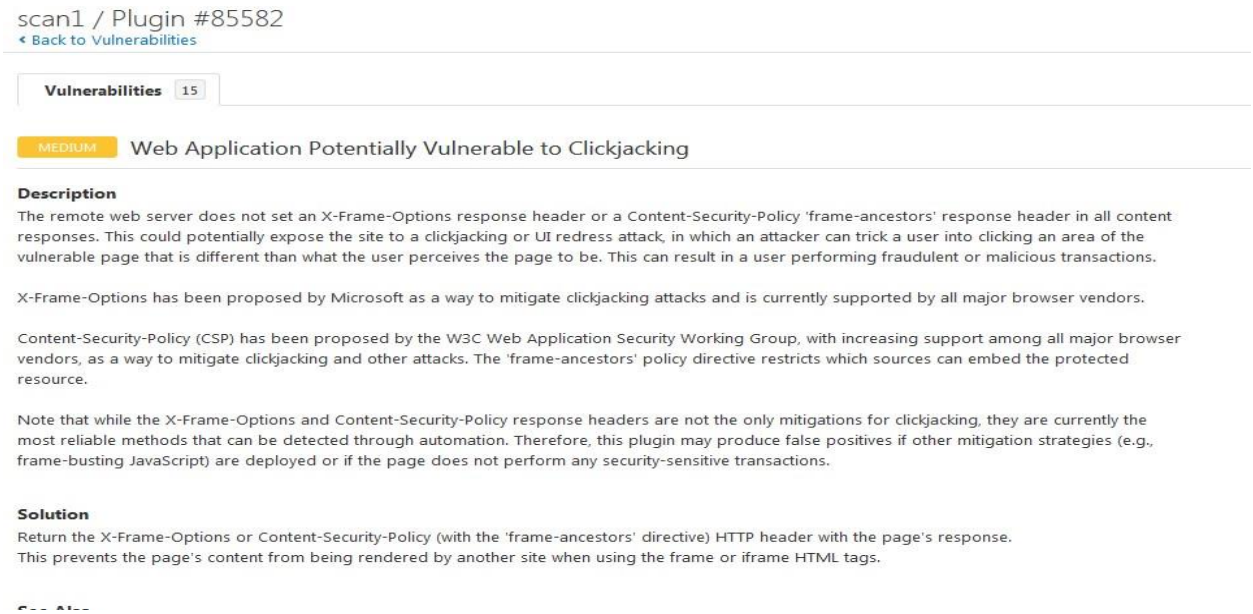

Рисунок 3.38 – Обнаруженная сканером уязвимость Clickjacking

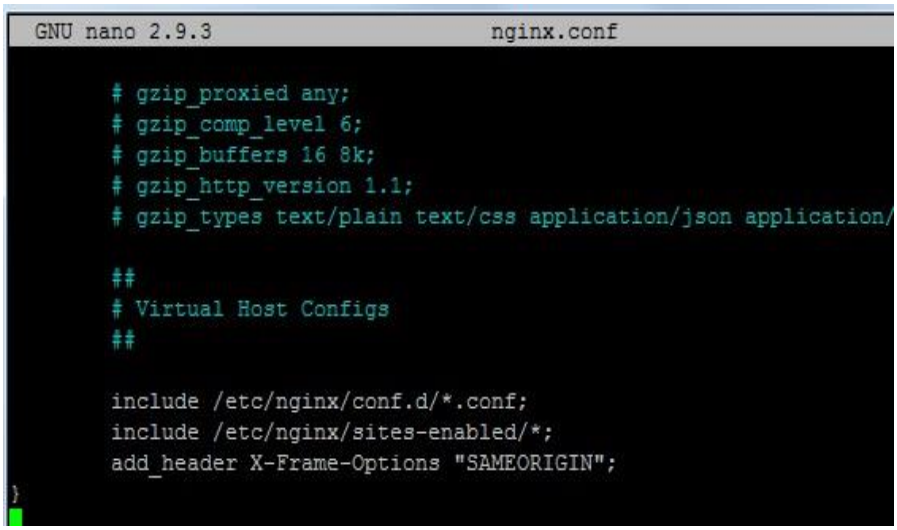

Рисунок 3.39 – Включение опций X-Frame-Options

Рисунок 3.40 – Заголовок ответа до включения опций

Рисунок 3.41 – Заголовок ответа после включения опций

Сканер обнаружил, включенный XML-RPC (рисунок 3.42). XML-RPC – это протокол удаленных процедур, который позволяет удаленно взаимодействовать с вашим сайтом WordPress. Другими словами, это способ управлять сайтом без необходимости входа в систему вручную через стандартную страницу «wp-login.php». В сообществе безопасности WordPress было много вопросов о XML-RPC. В основном есть две проблемы:

- xml-rpc можно использовать для DDoS-атаки;
- его можно использовать для повторного использования комбинаций имени пользователя и пароля.

Для отключения этого протокола используем плагин “disable xml rpc” (рисунок 3.43) и запретим доступ в конфигурационном файле (рисунок 3.44). Результат повторного сканирования можно увидеть на рисунке 3.45.

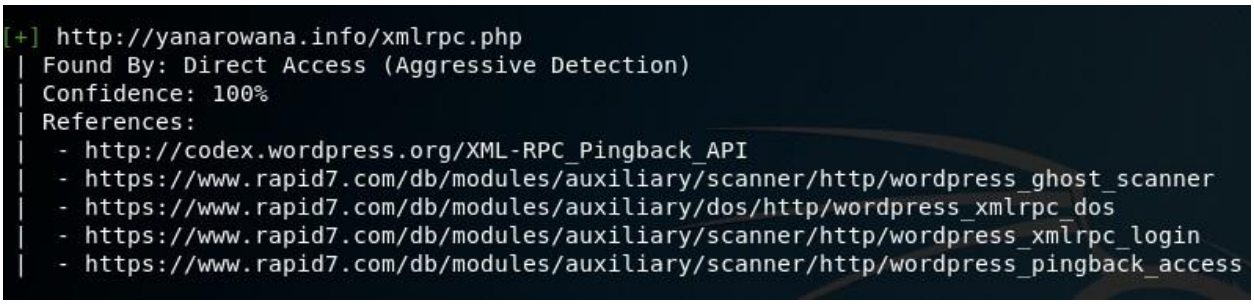

Рисунок 3.42 – Результат сканирования протокола xml-rpc

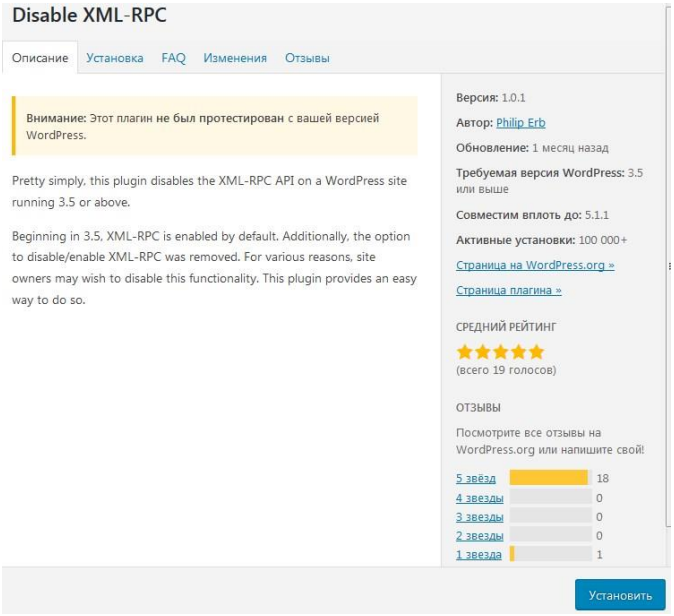

Рисунок 3.43 – Установка плагина Disable XML-RPC

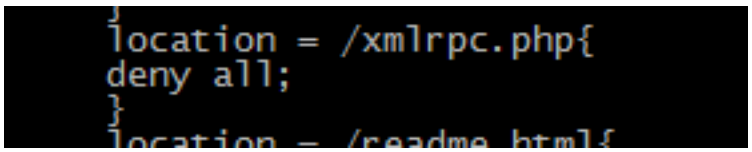

Рисунок 3.44 – Запрет обращению к файлу xmlrpc.php

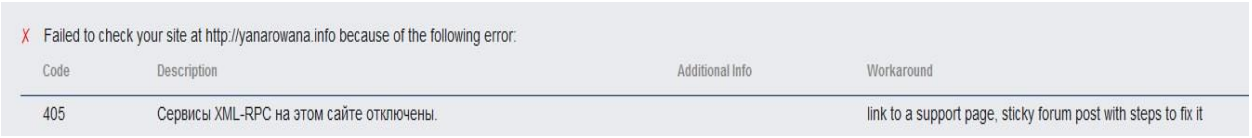

Рисунок 3.45 –Повторное сканирование на наличие протокола XML-RPC

Далее изменим некоторые параметры php.ini. Первым делом изменим значение cgi.fix_pathinfo на 0 (рисунок 3.46), ведь настройка по умолчанию очень небезопасна, потому что благодаря ей PHP попытается исполнить ближайший файл, который сможет найти в случае, когда запрашиваемый PHP файл не может быть найден. Это позволит пользователям сформировать PHP запросы таким образом, чтобы запускать скрипты, к которым у них не должно быть доступа.

Параметр session.cookie_lifetime устанавливаем на 0 (рисунок 3.47). 0 имеет особое значение. Он сообщает браузеру не сохранять cookie в постоянное хранилище. Следовательно, когда браузер закрывается, сессионные cookie сразу же удаляются. Если задать значение отличное от 0, это может позволить другим пользователям использовать эти cookie.

Включаем опцию session.use_strict_mode (рисунок 3.48). Это не позволит сессионному модулю использовать неинициализированные

идентификаторы сессий. Другими словами, сессионный модуль будет принимать только корректные идентификаторы, сгенерированные им же и будет игнорировать идентификаторы созданные на стороне пользователя.

Из-за особенностей спецификации cookie, атакующий может сделать cookie с идентификатором сессии неудаляемой с помощью локальной базы cookie или JavaScript-инъекцией. session.use_strict_mode может не дать атакующему использовать этот идентификатор.

Включаем опцию session.cookie_httponly (рисунок 3.49). Эта опция запрещает доступ к сессионной cookie для JavaScript. Эта опция предотвращает кражу cookie с помощью JavaScript-инъекции.

Включаем опцию session.cookie_secure (рисунок 3.50). Она разрешает получать доступ к cookie идентификатора сессии только при использовании протокола HTTPS. Если ваш сайт использует только протокол HTTPS, вам необходимо включить эту опцию.

В опции disable_function передаем значение всех потенциально опасных функции (рисунок 3.51).

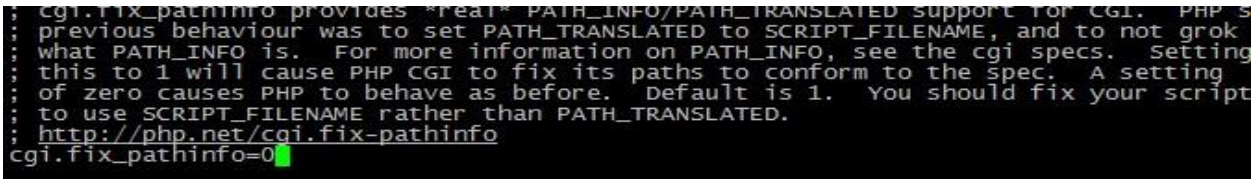

Рисунок 3.46 – Настройка параметра cgi.fix_pathinfo

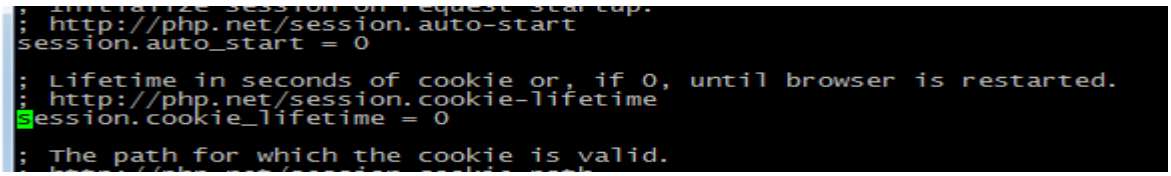

Рисунок 3.47 –Настройка параметра session.cookie_lifetime

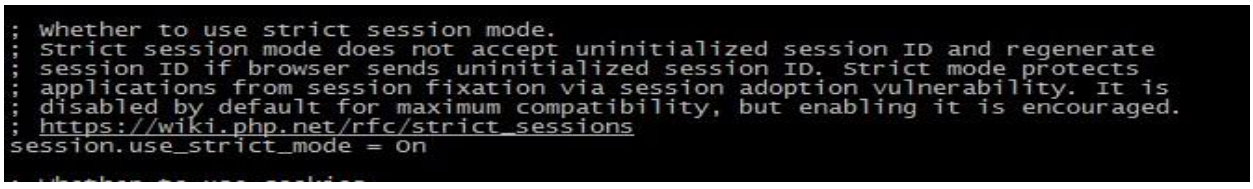

Рисунок 3.48 –Настройка параметра session.use_strict_mode

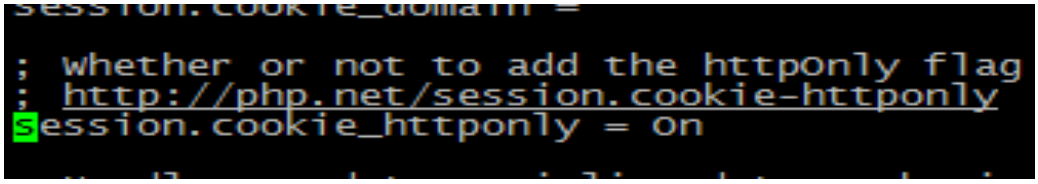

Рисунок 3.49 –Настройка параметра session.cookie_httponly

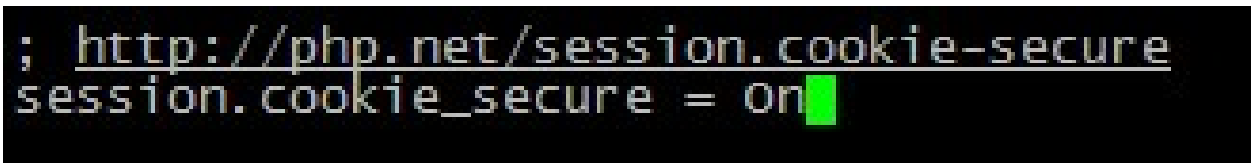

Рисунок 3.50 –Настройка параметра session.cookie_secure

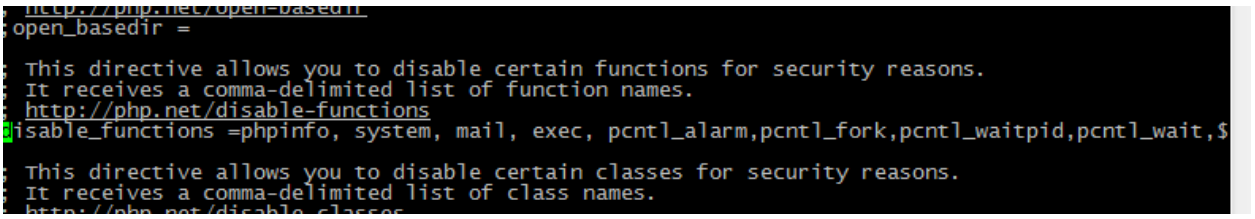

Рисунок 3.51 –Настройка параметра disable_functions

При повторном сканировании можно увидеть, что уязвимости устранены (рисунок 3.52):

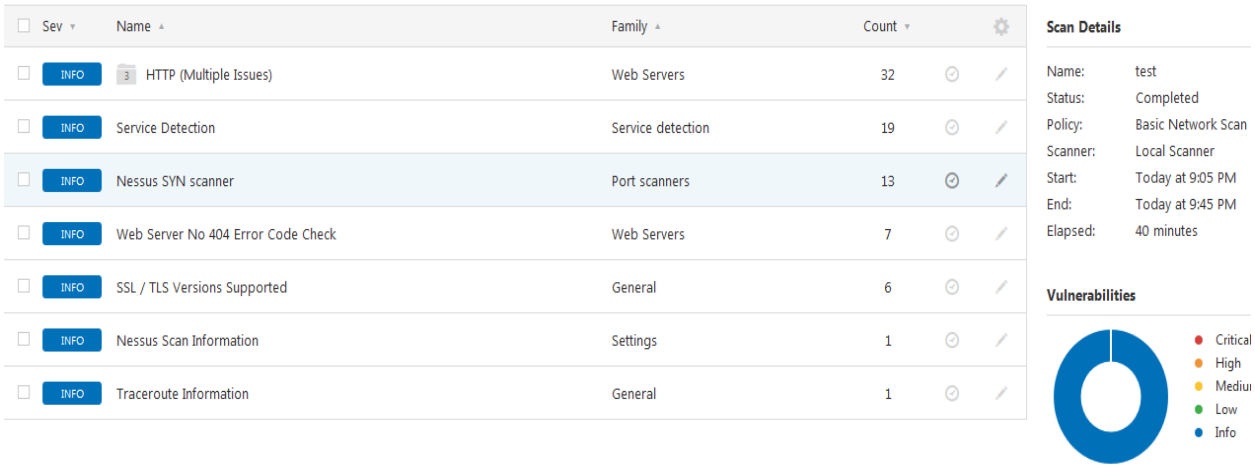

Рисунок 3.52 – Результат повторного сканирования Nessus

3.3 Настройка и обеспечение безопасности почтового сервера

Устанавливаю iRedMail, с помощью оболочки bash. Итоговые параметры установки представлены на рисунке 3.53.

Рисунок 3.53 – Параметры установки iRedMail

Для успешной доставки писем добавляю запись SPF в панели управления DNS записями (рисунок 3.54). Копирую открытый ключ DKIM из файла iRedMail.tips (рисунок 3.55). Добавляю его в содержимое записи DKIM в той же панели (рисунок 3.56). Задаю политику DMARC (рисунок 3.57). Результат тестирования SPF, DKIM, DMARC представлен на рисунке 3.58.

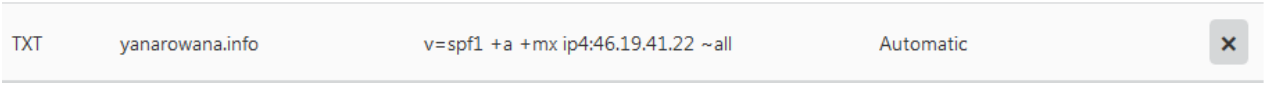

Рисунок 3.54 – Запись SPF на панели управления DNS записями

Рисунок 3.55 – Содержание файла iRedMail.tips

Рисунок 3.56 – Запись DKIM на панели управления DNS записями

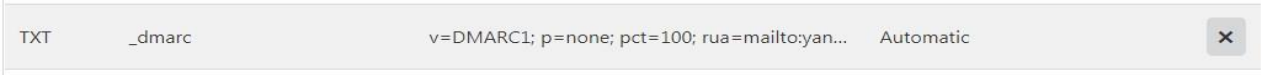

Рисунок 3.57 – Запись DMARC на панели управления DNS записями

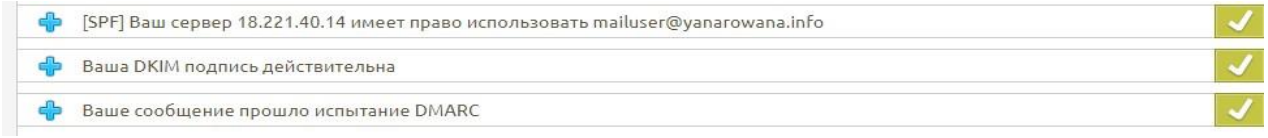

Рисунок 3.58 – Результат тестирования SPF, DKIM, DMARC

Для борьбы со спамом был отредактирован конфигурационный файл postfix. На рисунке 3.59 представлены правила для блокировки писем, если у клиента SMTP нет записи PTR, когда имя хоста HELO / EHLO не имеет A записи и не MX записи в DNS, имя хоста клиента SMTP не имеет действительной A записи. Были добавлены правила для блокировки писем, перечисленных в публичных черных списках (рисунок 3.60).

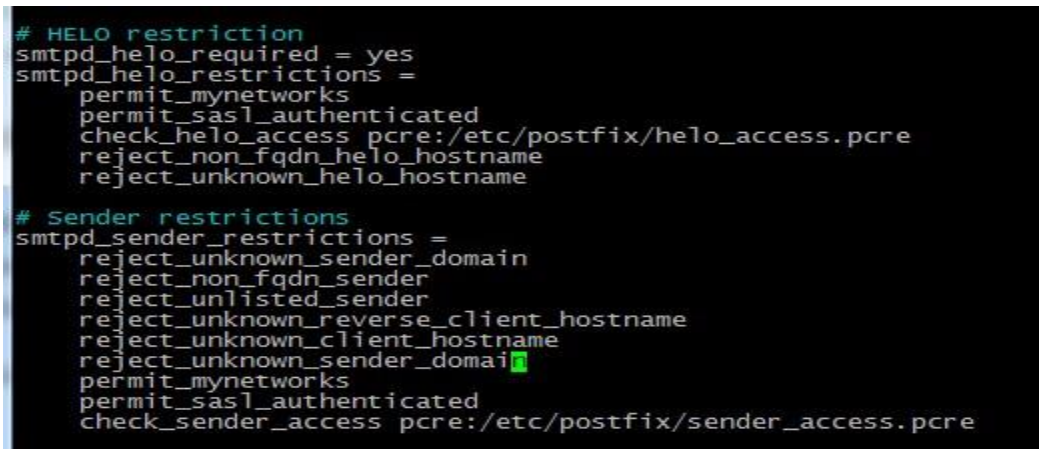

Рисунок 3.59 – Правила блокировки спама

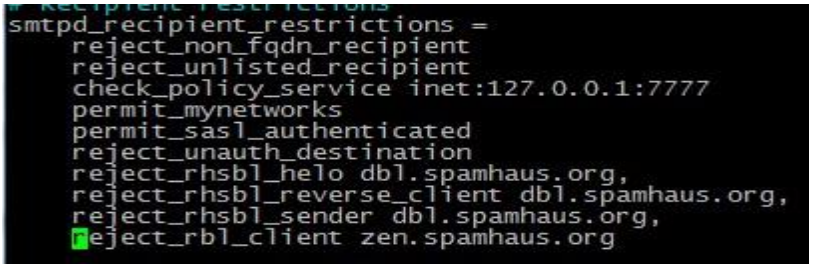

Рисунок 3.60 – Правила блокировки писем, перечисленных в публичных черных списках

Кроме этого, для борьбы со спамом, была настроена проверка заголовков и тел входящих писем. Для этого, первым делом, я включил параметры для проверки заголовков и содержимого (рисунок 3.61). Были добавлены правила блокировки писем на основе регулярных выражении (рисунок 3.62).

Рисунок 3.61 – Параметры для проверки содержимого письма

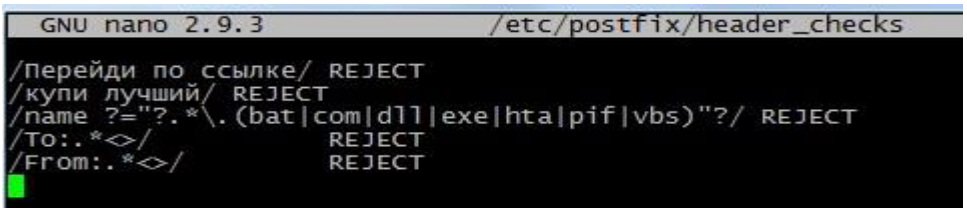

Рисунок 3.62 – Правила блокировки писем на основе регулярных выражении

Для более эффективной борьбы со спамом был настроен Spamassassin – были добавлены пользовательские правила проверки входящих писем (рисунок 3.63), настроены правила для отклонения приема письма, если то набирает 7 баллов Spamassasin (рисунок 3.64).

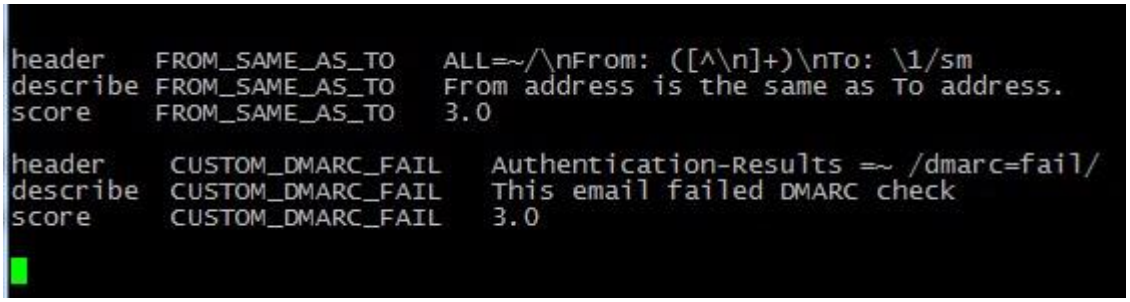

Рисунок 3.63 – Пользовательские правила Spamassasin

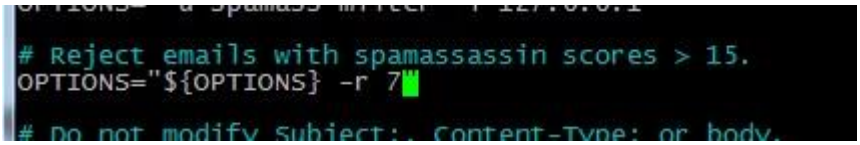

Рисунок 3.64 – Правила для отклонения писем

Вывод

В данной главе было продемонстрирована защита WEB и почтовых серверов. Сначала, был продемонстрирован процесс создания инстанса Amazon EC2 – выбор типа инстанса, настройка группы безопасности AWS, создание пары ключей SSH, для безопасной авторизации. Затем я безопасно развернул WEB-сервер Nginx, собрав его из исходного кода. Вместе с Nginx был установлен WAFnaxsi. Я активировал его, задав некоторые правила блокировки. Благодаря Cloudflare CDN я решил множество проблем безопасности – установил TLS сертификат, обеспечил защиту от DDoS-атак и защиту от сканеров каталогов. Для обеспечения удобного создания, редактирования и управления содержимым WEB-приложения была установлена и настроена Wordpress CMS. Используя плагины Wordpress, я добился еще одного уровня защиты. WAFWordfence, установленный в виде плагина на Wordpress помог избавиться от распространенных атак на WEB-приложения и установил надежную защиту от перебора паролей. После этого были проделаны стандартные действия для обеспечения безопасности WEB-сервера – изменен стандартный адрес входа в админ панель, изменен префикс таблиц Wordpress. Используя сканер уязвимостей, я нашел оставшиеся уязвимости и устранил их.

Во время настройки почтового сервера, я показал, как добиться того, чтобы исходящие письма были отправлены и не попадали в спам. Для этого необходимо было включить механизмы SPF, DKIM, DMARC. Задал некоторые правила для блокировки спама в конфигурационном файле postfix. Для более надежной защиты от спама использовал параметры проверки заголовков и текста писем, а так же Spamassassin для оценки письма.

4 Экономическая часть

4.1 Технико-экономическое обоснование

Данный дипломный проект посвящен изучению предметной области облачных платформ, в частности виртуальных серверов EC2, предоставляемые компанией Amazon на их собственной облачной платформе Amazon Web Services – AWS, WEB и почтовых серверов.

В разработке программного обеспечения будут участвовать: технический руководитель, отвечающий за координацию технического процесса и вопросы организационного характера, и программист-разработчик, в обязанности которого входит разработка технического обоснования, разработка, внедрение программного обеспечения, обеспечение его безопасности, тестирование и поддержка. Технико-экономическое обоснование содержит следующие пункты:

- определение сложности разработки программного обеспечения;
- расчет затрат на разработку ПО;
- определение ценности готового продукта;
- оценка результатов работы программного обеспечения.

4.2 Определение сложности разработки ПО

Для определения трудоемкости разработки модели приведен перечень всех основных этапов и видов работ, которые должны быть выполнены. Форма разделения работ по этапам с указанием трудоемкости их выполнения приведена в таблице 4.1

Таблица 4.1 – Этапы разработки ПО и трудоемкость

Этапы разработки ПО	Вид работы	Трудоемкость, чел. час.
1	2	3
Этап 1	Знакомство с материалами проекта	10
Этап 2	Изучение особенностей виртуальных серверов EC2	20
Этап 3	Изучение особенностей WEB сервера nginx	15
Этап 4	Изучение особенностей почтового сервера postfix	15
Этап 5	Развертывание и настройка WEB сервера nginx в сервере EC2, работающего на базе Ubuntu Sever 18.0	30
Этап 6	Обеспечение безопасности WEB сервера nginx в сервере EC2, работающего на базе Ubuntu Sever 18.04	30
Этап 7	Развертывание и настройка почтового сервера postfix в сервере EC2, работающего на базе Ubuntu Sever 18.04	30

1	2	3
Этап 8	Обеспечение безопасности почтового сервера postfix в сервере EC2, работающего на базе Ubuntu Sever 18.04	30
Этап 9	Анализ выполненных работ	20
Итого: трудоемкость выполнения проекта		200

4.3 Расчет затрат на разработку ПО

Определение затрат необходимых для разработки программного обеспечения производится на основе имеющейся сметы, которая включает следующие элементы:

- материальные затраты;
- затраты на оплату труда;
- социальный налог;
- амортизация основных фондов;
- прочие затраты.

Материальные затраты делятся на основные и вспомогательные затраты на материалы, энергию и другие затраты необходимые для разработки ПО. Расчет материальных затрат происходит по форме, предоставленной в таблице 4.2.

Таблица 4.2 – Затраты на материальные ресурсы

Наименование материала	Марка	Ед. измерения	Количество	Цена за ед. в тенге	Сумма в тенге
Бумага для офиса	Снегурочка	Упаковка	3	1 000	3 000
Тетрадь (96 листов)	Abdi	Штук	2	200	400
Ручки	Cello	Штук	2	70	140
Итого:					3 540

Общую сумму, необходимую на материальные средства ($З_м$) можно рассчитать по следующей формуле:

$$З_м = \sum P_i * Ц_i, \quad (4.1)$$

где P_i - расход i-го вида материального ресурса, натуральные единицы;

$Ц_i$ - цена за единицу i-го вида материального ресурса, тг;

i - вид материального ресурса;

n - количество видов материальных ресурсов.

Расчет затрат на необходимое оборудование и программное обеспечение производится по форме, приведенной в таблице 4.3.

Таблица 4.3 – Расчет затрат на оборудование и ПО, необходимое для проекта

Наименование материала	Марка	Ед. измерения	Количество	Цена за ед. в тенге	Сумма в тенге
Системный блок	Hp p6-2059ci	Штук	1	130 000	130 000
Монитор	Qmax M901B	Штук	1	21 000	21 000
Мышь	Hp 200	Штук	1	2000	2000
Клавиатура	Hp QY776A	Штук	1	2000	2000
Принтер	Hp LaserJet-1018	Штук	1	20 000	20 000
Маршрутизатор	MikroTik hEX S RB760iGS	Штук	1	30 000	30 000
ОС	Windows 10	Штук	1	-	-
Домен	PS.kz	Штук	1	1 100	1 100
Итого:					206 100

Зм = 3 540 + 206 100 = 209 640 (тг)

Для реализации программного обеспечения необходимы материалы на сумму 209 640 тенге.

4.4 Расчет затрат на электроэнергию

Так как при разработке программного обеспечения не обойтись без потребления электроэнергии, имеет смысл произвести расчет затрат на электроэнергию.

Время работы оборудования для разработки ПО равно 200 часам, данное количество часов было рассчитано в таблице 4.1. Для принтера расчет будет проводиться для периода в 24 часа, так как нет необходимости в постоянной эксплуатации принтера.

$$Э = З_{эл.эн.обор.} + З_{доп.нужды.}, \quad (4.2)$$

где $З_{эл.эн.обор.}$ – затраты на электроэнергию оборудования;

$З_{доп.нужды.}$ – затраты электроэнергии на дополнительные нужды.

Расчет электроэнергии, которая необходима для оборудования определяется по следующей формуле:

$$З_{эл.эн.обор.} = \sum W * К_{исц} * S * T, \quad (4.3)$$

где W – потребляемая мощность, Вт;

$К_{исц}$ – коэффициент использования ($К_{исц}$ = 0,7..0,9);

T – время работы;

S – тариф (1кВт/ч = 18,32 тг).

Итоги по расчетам стоимости затрачиваемой электроэнергии представлены в таблице 4.4.

Таблица 4.4 – Затраты на электроэнергию

Наименование приборов	Паспортная мощность, кВт	Коэффициент мощности	Время работы оборудования, ч	Цена ЭЭ тг/кВтч	Сумма, тг.
Системный блок	0,3	0,9	200	18,32	989,28
Монитор	0,05	0,9	200	18,32	164,88
Модем	0,02	0,9	200	18,32	65,95
Принтер	0,25	0,7	24	18,32	76,94
Кондиционер	2,2	0,9	200	18,32	7254,72
Освещение	0,3	0,7	200	18,32	769,44
Итого:					9321,21

$$З_{эл.эн.обор.} = 9321{,}21 \text{ (тенге)}$$

На дополнительные потребности расходы подсчитываются на основе повышенного показателя в объеме 5% от расходов на электроэнергию:

$$З_{доп.нужды} = 5\% * З_{эл.эн.обор.}, \quad (4.4)$$

Определим затраты на дополнительные потребности согласно формуле (4.4):

$$З_{доп.нужды} = 0.05 * 9321{,}21 = 466{,}06 \text{ (тенге)}$$

Исходя из всех расчетов, полные расходы на электроэнергию составляют:

$$Э = 466{,}06 + 9321{,}21 = 9787{,}27 \text{(тенге)}$$

4.5 Расчет затрат на оплату труда

За реализацию проекта, как описывалось ранее, будут отвечать технический руководитель проекта и программист-разработчик.

Сумму расходов на оплату труда можно рассчитать по следующей формуле:

$$З_{тр} = \sum ЧС_i * T_i \quad (4.5)$$

где $ЧС_i$ - часовая ставка i-го работника, тг;

T_i - трудоемкость разработки модели, чел.×ч; i - категория работника;

n - количество работников, занятых разработкой ПП.

Во время реализации проекта рабочее время участников не равномерно, поэтому имеет смысл установить часовую ставку каждого работника и общий объем заработной платы.

Часовую ставку сотрудника можно рассчитать по следующей формуле:

$$ЧС_i = \frac{ЗП_i}{ФРВ_i} \quad (4.6)$$

где $ЗП_i$ - месячная заработная плата i-го работника, тг;

$ФРВ_i$ - месячный фонд рабочего времени i-го работника, час.

Месячная заработная плата технического руководителя равняется 200 000 тенге, а программиста-разработчика – 170 000 тенге. Рассчитаем часовую ставку каждого работника согласно формуле (4.6).

Для технического руководителя:

$$ЧС_{техрук} = \frac{200\,000}{22 * 8} = 1\,136{,}4 \text{ тг/ч}$$

Для программиста-разработчика:

$$ЧС_{разр} = \frac{170\,000}{22 * 8} = 965{,}9 \text{ тг/ч}$$

Часовая ставка руководителя составляет 1 136,4 (тг/ч), трудоемкость разработки равняется 90 часам. Часовая ставка разработчика составляет 965,9 (тг/ч), трудоемкость разработки равняется 200 часам. Согласно формуле (4.5) можно рассчитать сумму расходов на заработную плату работников:

$$З_{тр} = 1136{,}4 * 90 + 965{,}9 * 200 = 102\,276 + 193\,180 = 295\,456$$ (тенге)

Расчеты затрат по оплате труда показаны в таблице (4.5).

Таблица 4.5. – Расчет заработной платы

Категория работника	Квалификация	Трудоемкость разработки ПП, час.	Часовая ставка, тг/ч	Сумма, тг.
Руководитель	Руководитель проекта	90	1 136,4	102 276
Разработчик	Программист	200	965,9	193 180
Итого:				295 456

4.6 Расчет затрат по социальному налогу

Согласно Налоговому кодексу Республики Казахстан социальный налог составляет 9,5% от фонта оплаты труда. Социальный налог можно рассчитать по следующей формуле:

$$С_н = (ФОТ - ПО) * 0{,}095 \quad (4.7)$$

где ПО - отчисления в пенсионный фонд, они составляют 10% от ФОТ.

$$ПО = 295\,456 * 0{,}1 = 29\,545{,}6 \text{ тенге}$$
$$С_н = (295\,456 - 29545{,}6) * 0{,}095 = 25\,261{,}5 \text{ тенге}$$

Результаты расчетов представлены в таблице (4,6):

Таблица 4.6 – Начисление социального налога

Категория работника	Количество человек	Заработная плата, тг	Пенсионные отчисления, тг	Социальный налог, тг
Руководитель	1	102 276	10227,6	8744,6
Разработчик	1	193 180	19318	16516,9
Итого:				25 261,5

4.7 Амортизация основных фондов и прочие затраты

Нормы амортизации ОФ необходимо определить в соответствии с налоговым кодексом РК. Амортизацию ОФ можно определить по следующей формуле:

$$A_г = \frac{С_{об} * Н_а}{100} \quad (4.8)$$

где, $С_{об}$ – стоимость оборудования;

$Н_а$ – норма амортизации (норма амортизация = 25);

Формула (4.8) позволяет рассчитать нужную сумму для амортизационных отчислений за год для системного блока :

$$A_г = \frac{130\,000 * 25}{100} = 32\,500 \text{ тенге}$$

Теперь необходимо рассчитать норму амортизации за период разработки:

$$A_г = \frac{32\,500 * 34}{365} = 3\,027{,}4 \text{ тенге}$$

Подобным образом необходимо рассчитать норму амортизации для всего оборудования. Результаты расчетов приведены в таблице (4.7).

Таблица 4.7 – Амортизация ОФ

Наименование оборудования и ПО	Стоимость оборудования и ПО, тг	Годовая норма амортизации, %	Сумма амортизации за год, тг	Сумма амортизации за время разработки, тг
Системный блок	130 000	25	32 500	3 027,4
Монитор	21 000	25	5 250	489
Мышь	2000	25	500	46,6
Клавиатура	2000	25	500	46,6
Принтер	20 000	25	5 000	466,5
Маршрутизатор	30 000	20	6 000	558,9
Домен	1 100	15	165	15,4
Итого:			49 915	4 650,4

Смета расходов на разработку ПО.

На основе всех представленных расчетов необходимо оформить смету расходов на разработку ПО согласно форме, которая приведена в таблице (4.8). На рисунке 1 продемонстрирована диаграмма рабочих расходов.

Таблица 4.8 – Смета затрат на разработку ПО

Статьи затрат	Сумма, тг
Затраты на оборудование и материальные ресурсы	209 640
Затраты на оплату труда	295 456
Социальные налоги	25 261,5
Затраты на электроэнергию	9787,27
Амортизация основных фондов	4 650,4
Итого по смете:	544 795,17

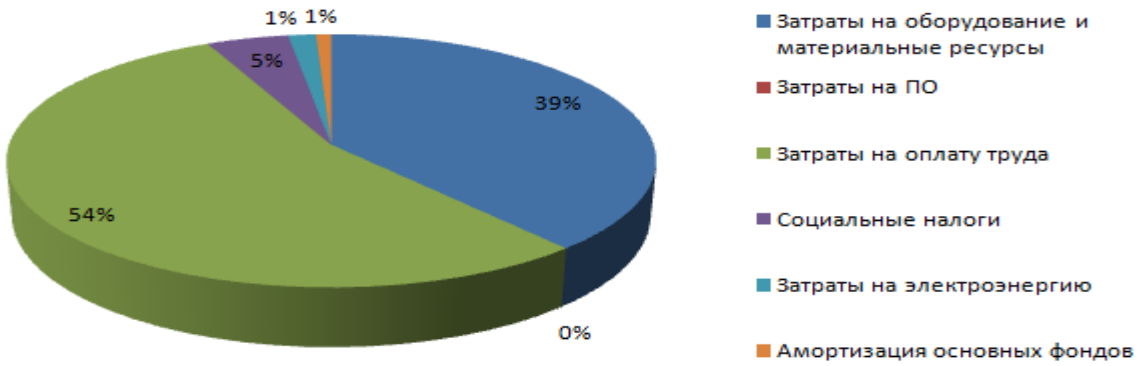

Рисунок 4.1 – Диаграмма затрат

4.8 Определение возможной (договорной) цены ПО

Стоимость программного обеспечения определяется на основе качества разработанного продукта, сроков его разработки и производительности продукта. Стоимость $Ц_Д$ для программного обеспечения можно рассчитать по следующей формуле:

$$Ц_Д = З_{нир}\left(1 + \frac{P}{100}\right), \quad (4.9)$$

где $З_{нир}$ – затраты на разработку программного обеспечения, тг;

Р – средний уровень рентабельности ПО, (%). Данный параметр принят равным 25%.

$$Ц_Д = 544\,795{,}17\left(1 + \frac{25}{100}\right) = 680\,993{,}96 \text{тенге}$$

Далее необходимо определить стоимость реализации с учетом НДС, ставка НДС устанавливается законодательством РК. На 2019 года ставка НДС составляет 12%. Стоимость реализации, учитывая НДС можно рассчитать по следующей формуле [15]:

$$Ц_р = Ц_д + Ц_д * НДС, \quad (4.10)$$

$$Ц_р = 680\,993{,}96 + 680\,993{,}96 * 0{,}12 = 762\,713{,}235 \text{ тенге}$$

Данную цену можно округлить до 763 000 тенге.

Вывод

Данная глава дипломного проекта содержит экономические расчеты, которые позволяют определить затраты необходимые для разработки программного продукта. Расчеты включают в себя:

- расчет трудоемкости разработки программного продукта;
- расчет затрат на разработку программного продукта;
- расчет затрат на электроэнергию;
- расчет затрат на оплату труда;
- расчет затрат по социальному налогу;
- амортизация основных фондов и прочие затраты.

Договорная цена программного продукта равняется 763 000 тенге, данное значение является рациональным с точки зрения экономической эффективности.

5 Безопасность жизнедеятельности

5.1 Анализ условий труда

Развертывание и последующая организация защиты WEB и почтовых серверов осуществляется с использованием компьютерного оборудования и программного обеспечения. Для продуктивного рабочего процесса работника или работников необходимо обеспечить его или их оптимальными условиями для работы: удобными стульями или креслами, снижающих нагрузку на позвоночник; в меру просторным рабочим пространством, которое позволит работнику проделывать все необходимые действия и перемещения; аспирационными системами, обеспечивающие вентиляцию воздуха и поддерживающие комнатную температуру на рабочем месте.

5.2 Описание рабочего места и используемого оборудования

Рабочее помещение расположено на 2 этаже бизнес центра. Планировка помещения представлена на рисунке 1.

В рабочем помещении трудятся 3 сотрудника (мужчины – 2, женщины – 1), включая меня, которые имеют служебные места.

Характеристики кабинета: длина L = 6 метров, ширина W= 4 метров, высота H = 3 метра. Присутствует окно площадью 2 м^2. Установлен старый кондиционер Daikin.

Для работы используются системные блоки, мониторы и принтеры – 3шт. Технические характеристики:

- Intel® Pentium® G630;
- Vectron H110;
- Transcend TS512MLK72V1N-4GB DDR3 1066MHz;
- HDD Toshbida DT01ACA100 100GB;
- FSP Group Q-Dion QD-400Z (9PA300AQ25) 300W;
- Qmax M901B 18.5";
- Hp LaserJet 1018.

Оборудования не представляют шумовую угрозу. Анализ условий труда показал, что слабым местом является вентиляция, в связи с этим в данном разделе производится расчет искусственной вентиляции.

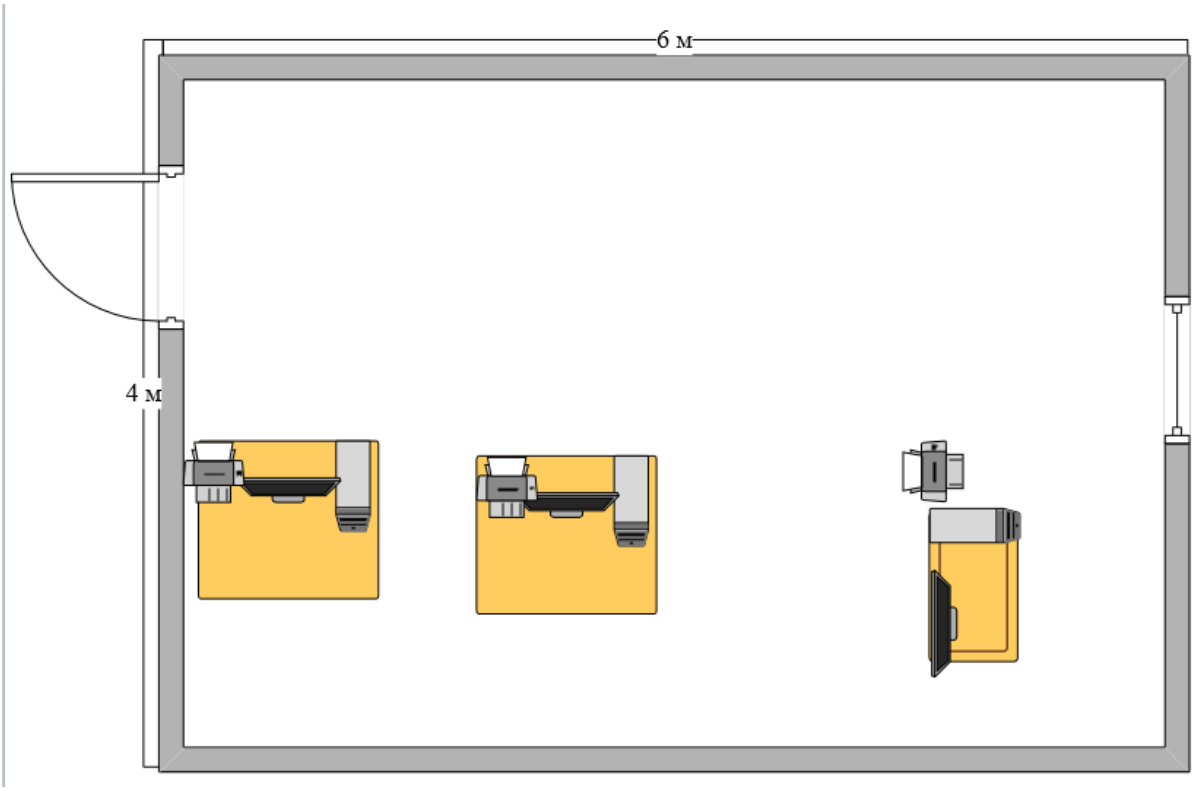

Рисунок 5.1 – Планировка рабочего помещения

5.3 Расчет

В помещениях различного назначения действуют в основном тепловые нагрузки, возникающие снаружи помещения (наружные); а также тепловые нагрузки, возникающие внутри зданий (внутренние).

В помещениях различного назначения действуют в основном тепловые нагрузки, возникающие снаружи помещения (наружные); а также тепловые нагрузки, возникающие внутри зданий (внутренние).

Наружные тепловые нагрузки

В зависимости от времени года и времени суток наружные тепловые нагрузки могут быть положительными. Теплопоступления и теплопотери в результате разности температур определяются по формуле 0.1:

$Q_{огр} = V_{пом} * X_о * (t_{Нрасч} – t_{Врасч})$, Вт (0.1), где

$V_{пом}$ – объем помещения, $м^3$;

$V_{пом} = 6*4*3=72$ $м^3$;

$X_о$ – удельная тепловая характеристика, $Вт/м^{3}*^{0}С$;

$X_о= 0,42$ $Вт/м^{3}*^{0}С$;

$t_{Нрасч}$ – наружная температура (параметр А). Для холодного периода – средняя температура самого холодного месяца в 14 часов, для теплого периода – средней температуре самого жаркого месяца в 14 часов.

$t_{Врасч}$ – внутренняя температура, выбирается с учетом комфортных условий или технологических требований, предъявляемых к производственным процессам.

Для теплого времени года:

$t_{Нрасч} = 29{,}4\ ^0C$

$t_{Врасч} = 26\ ^0C$

$Q_{огр}= 72*0{,}42\cdot 3{,}4=102{,}8$ Вт

Для холодного времени года

$t_{Нрасч} = -9\ ^0C$

$t_{Врасч} = 19\ ^0C$

$Q_{огр} = 72\cdot 0{,}42\cdot 28 = 846{,}7$ Вт

Избыточная теплота солнечного излучения в зависимости от типа стекла почти до 90% поглощается средой помещения, остальная часть отражается. Максимальная тепловая нагрузка достигается при максимальном уровне излучения, которое имеет прямую и рассеянную составляющие. Интенсивность излучения зависит от ширины местности, времени года и времени суток.

Теплопоступление от солнечного излучения через остекление определяется по формуле 0.2:

$Q_р = (q^I F^I_o + q^{II} F^{II}_o) * \beta с.з$ (0.2), где

q^I, q^{II} – тепловые потоки от прямой и рассеянной солнечной радиации, Вт/м^2;

F^I_0, F^{II}_0 – площади светового проема, облучаемые и необлучаемые прямой солнечной радиацией, м^2 ;

βс.з. – коэффициент теплопропускания. Для штор-жалюзи с металлическими пластинами:

βс.з. = 0,15

При отсутствии наружных затеняющих козырьков, ребер и т. д. для периода облучения остекления солнцем, когда его лучи проникают через окно в помещение $F^I_0 = F^{II}_0 = F_0 = 0$:

$Q_р = q^{II} F_0 * \beta с.з = q_{вр} * K^T_1 * K_2 * \beta с.з * n * S_0$ (0.3), где

$q_{вп}$; $q_{вр}$ – тепловые потоки от рассеянной радиации, Вт/м^2. Для широты в 44^0 СШ после полудня в 14-15 ч. при расположении ЮВ:

$q_{вр} = 63$ Вт/м^2;

$F_o = nS_o = 1*2 = 2$ м^2 – площадь светового проема (n – число окон; S_o – площадь 1 окна);

K_1 – коэффициент затемнения остекления переплетами (K^T_1 – для проемов в тени).

$K^T_1 = 1{,}28$;

K_2 – коэффициент загрязнения остекления:

$K_2 = 0{,}95$.

Тогда:

$Q_р = 63*1{,}28*0{,}95*0{,}15*2=22{,}9$ Вт

Для широты в 44^0СШ после полудня в 14-15 ч. при расположении ЮЗ:

$q_{вр} = 101$ Вт/м^2;

$F_o = nS_o = 1*2=2$ м^2

Тогда:

$Q_р = 101*1,28*0,95*0,15*2=36,8$ Вт

Тогда общее теплопоступление солнечного излучения с обеих сторон равно:

$Q_р = 22,9+36,8 = 59,7$ Вт

Внутренние тепловые нагрузки

Внутренние нагрузки в жилых, офисных или относящихся к сфере обслуживания помещениях слагаются в основном из тепла:

- выделяемого людьми;
- выделяемого лампами и осветительными, электробытовыми приборами;
- выделяемого компьютерами, печатающими устройствами фотокопировальными машинами пр.;

Летом при 24 ^{0}С один мужчина выделяет явного тепла 67 Вт, а общего – 102 Вт. Женщина выделяет 85% от нормы тепловыделений взрослого мужчины. Тогда выделение явного тепла в помещении составит:

$Q_л^я = 67*2 + 67*1*0,85 = 190,95$ Вт

А выделение общего тепла:

$Q_л^о = 102*2 + 102*1*0,85 = 290,7$ Вт

Зимой при 18 ^{0}С один мужчина выделяет явного тепла 89 Вт, а общего – 104 Вт. Тогда выделение явного тепла в помещении составит:

$Q_з^я = 89*2 + 89*1*0,85= 253,65$ Вт

А выделение общего тепла:

$Q_з^о = 104*2 + 104*1*0,85= 296,4$ Вт

Теплопоступление от осветительных приборов, оргтехники и оборудования рассчитывается следующим образом. Теплопоступление от ламп определяется по формуле:

$$Q_{осв} = \eta \cdot N_{осв} \cdot F_{пол}, \quad Вт \qquad (0.4)$$

где η – коэффициент перехода электрической энергии в тепловую (для лампы накаливания η=0,92-0,97);

$N_{осв}$ – установленная мощность ламп (N=60 Вт/м^2);

$F_{пол}$ – площадь пола:

$F_{пол} = 6*4=24$ м^2

Тогда:

$Q_{осв} = 0,92*60*24 = 1324$ Вт

Тепло, выделяемое производственным оборудованием, определяется по формуле:

$$Q_{об} = N_{уст} \cdot K \qquad (0.5)$$

$Q_{об} = 0,3 * 3 * 0,75 * 10^3 = 0,67$ кВт.

Теплопритоки, возникающие за счёт находящейся оргтехники – это 30% мощности оборудования:

$Q_{орг} = 3 * 0,3 * 0,3 * 10^3 = 0,27$ кВт

Расчет теплового баланса помещения

На основании выполненных расчетов составим баланс теплопоступлений в помещении:

$$Q_{изб} = Q_р + Q^я + Q_{осв} + Q_{об} + Q_{орг} + Q_{огр}$$

Лето: $Q_{изб}^{Л} = 59,7 + 190,95 + 1324 + 670 + 270 + 102,8 = 2,61$ кВт

Зима: $Q_{изб}^{З} = 59,7 + 253,65 + 1324 + 670 + 270 + 846,7 = 3,42$ кВт

Так как тепловой баланс для лета больше зимнего теплового баланса, то рассчитаем теплонапряженность воздуха по формуле:

$$Q_н = \frac{Q_{изб.лето} \times 860}{V_{пом}}$$

$$Q_н = \frac{2,61 \cdot 860}{72} = 31,1 \text{ ккал/м3}$$

При $Q_н$ >20ккал/м^3, Δt =8 °С,

при $Q_н$ <20ккал/м^3, Δt =6 °С.

Определение количества воздуха, необходимое для поступления в помещение:

$$L = \frac{Q_{изб} \times 860}{C \times \Delta t \times \gamma}$$

$$L = \frac{0,34 \cdot 860}{0,24 \cdot 8 \cdot 1,206} = 128,3 \text{ м3/час}$$

где С=0,24 ккал/(кг · °С) – теплоемкость воздуха,

γ=1,206 кг/м^3 – удельная масса приточного воздуха [14].

5.4 Выбор кондиционера. Схема расположения

Исходя из полученных результатов, для удаления лишнего тепла и очистки воздуха нужно использовать вентиляционную систему, которая способна обеспечить требуемую подачу воздуха L=128,3 (м3/ч). В данном случае подойдет Кондиционер MIDEA AURORA 1 MSAB-24HRN1-WG . Данный кондиционер способен обеспечить подачу воздуха до 1200 м3 /ч.

Технические характеристики:

- мощность (охлаждение): 2.2 кВт;
- мощность (обогрев): 2.2 кВт;
- потребляемая мощность при охлаждении: 2100 Вт;
- потребляемая мощность при обогреве: 2330 Вт;
- обслуживаемая площадь: 30 м2;
- уровень шума внутреннего блока: 37-41 дБ;
- уровень шума внешнего блока: 50 дБ;
- цвет: белый.

Характеристики подключения:

- вентиляция: 1200 м3/час;
- класс энергоэффективности при охлаждение/обогреве: А++/А+;
- электропитание, В/Гц/Ф:220 Вт;
- энергопотребление в режиме ожидания не более 1 Вт.

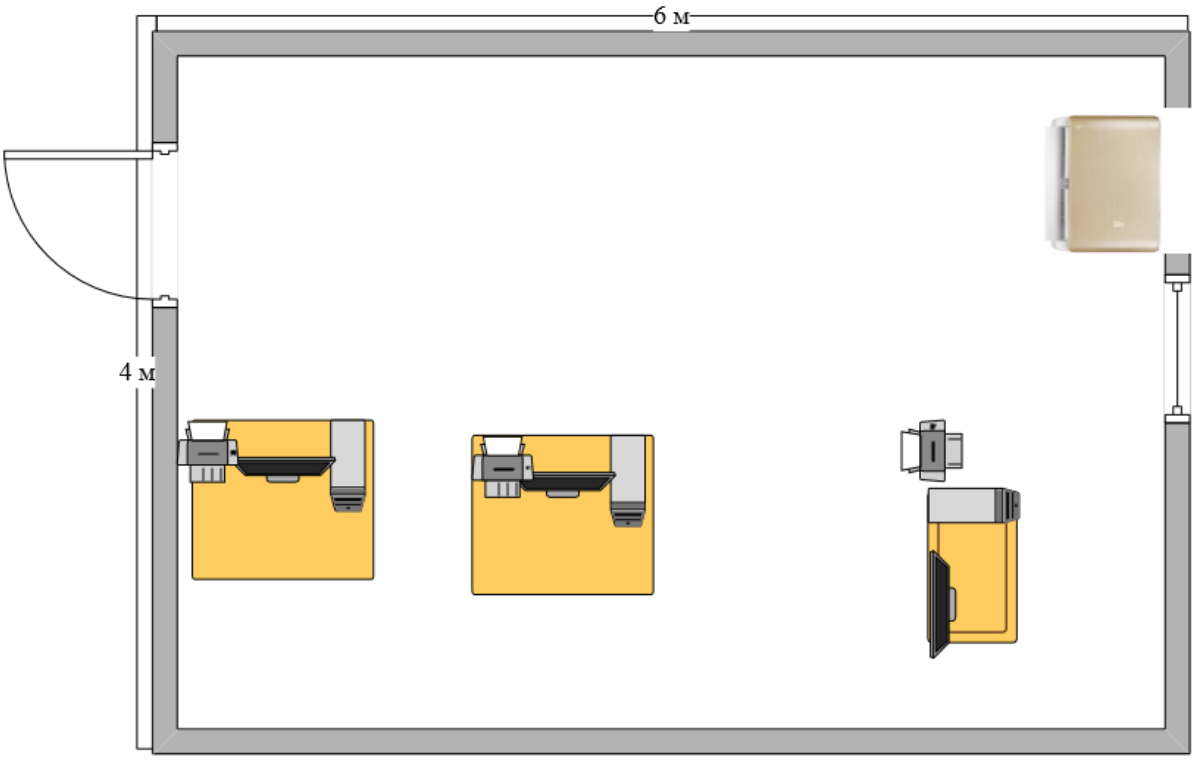

Рисунок 5.2 – Расположение кондиционера в помещении

Вывод

В этом разделе моего дипломного проекта я рассмотрел и рассчитал воздушные показатели для благоприятных условий труда, а именно, тепловые нагрузки в помещении, наружные и внутренние. По расчетам, для создания хороших условий труда необходим один кондиционер с подачей воздуха не менее 128,3 м3 /ч, в моем случае используется кондиционер MIDEA AURORA 1 MSAB-24HRN1-WG с подачей воздуха до 1200 м3 /ч.

Заключение

Сегодня трудно поспорить с тем, что технологии, основанные на облачных вычислениях, невероятно востребованы и активно развиваются. Под технологией облачных вычислений понимается технология, которая позволяет объединять ИТ-ресурсы различных аппаратных платформ в единое целое и предоставлять пользователю доступ к ним через сеть Интернет. Облачные сервисы предоставляют пользователям через сеть Интернет доступ к своим ресурсам посредством бесплатных или условно бесплатных облачных приложений, программные и аппаратные требования которых не предполагают наличия у клиентов высокопроизводительных компьютеров. Сегодня многие компании переносят свои проекты в облако, пользуясь различными облачными сервисами, и полностью доверяя защиту проектов провайдерам облачных сервисов. Но это далеко не панацея от всех угроз, которые могут нанести ущерб информационным активам. И WEB, и почтовые серверы, будучи ценными информационными активами, нуждаются в защите от атак. В своем дипломном проекте я развернул и обеспечил безопасность WEB и почтовых серверов на платформе облачных вычислении Amazon Web Services.

Для обеспечения защиты WEB-сервера я принял меры, описанные во второй главе настоящего дипломного проекта. WEB-сервер должен принимать подключения только по протоколам HTTP и HTTPS. Поэтому первым делом я задал правила в группах безопасности AWS. Задал правила, разрешающие потоку HTTP-трафика проходить через порт 80 с исходного адреса 0.0.0.0/0, то есть с любого места, а потоку HTTPS-трафика – через порт 443 с исходного адреса 0.0.0.0/0. Эти входящие правила разрешают трафик с адресов IPv4. Чтобы разрешить трафик IPv6, добавил входящие правила на те же порты с адреса источника :: / 0. Затем создал пару ключей SSH, для безопасного подключения к серверу. Уже на созданной виртуальной машине был развернут WEB-сервер Nginx, собранный из исходного кода, включающий в себя WAFnaxsi и заранее отключенные потенциально опасные модули. Для защиты от распространенных атак, типа sql-инъекции и xss-атак, были включены правила блокировки naxsi. Для обеспечения безопасного подключения и использования протокола HTTPS на сервере был установлен TLS сертификат, предоставленный Cloudfalre CDN. Также благодаря Cloudfalre CDN была включена защита от DDoS атак, и обеспечена защита от сканеров каталогов. Были изменены настройки по умолчанию конфигурационных файлов WEB-сервера для защиты от SlowLoris атаки.

Для обеспечения удобного создания, редактирования и управления содержимым WEB-приложения была установлена и настроена Wordpress CMS. Для создания дополнительного слоя защиты в Worfpress был установлен Wordfence, брандмауэр уровня приложения. Так же как и naxsi он защищает сайт от распространенных атак, типа sql-инъекции и xss-атак. Немало важной функцией Wordfence оказалась защита от перебора логинов и паролей. Эта

функция была настроена так, что после 20 попыток ввода неверного логина и пароля в течение 4 часов ip-адрес пользователя заблокируется на 4 часа. После этого были проделаны стандартные действия для обеспечения безопасности WEB-сервера: изменен стандартный адрес входа в админ панель, изменен префикс таблиц Wordpress.

С помощью сканера WEB уязвимостей были обнаружены и устранены уязвимости, типа clickjacking, wordrpress user enumeration, уязвимости, связанные с xml-rpc.php. Отредактировав файл php.ini, я обеспечил безопасность сессий и отключил выполнение потенциально опасных php функции.

После завершения работы с WEB-сервером, я развернул почтовый сервер. Я добился успешной и безопасной доставки исходящей почты, используя несколько механизмов. Сначала я добавил и правильно настроил SPF-запись, DKIM-запись в панели управления DNS записями. Я получил публичный ключ DKIM из файла настроек iRedMail.tips.

Для борьбы со спамом были использованы средства предоставляемые postfix и spamassasin. В конфигурационном файле postfix были заданы следующие правила:

- отклонить электронную почту, если у клиента SMTP нет записи PTR;
- включить ограничения HELO / EHLO Hostname в Postfix;
- отклонить электронную почту, если имя хоста клиента SMTP не имеет действительной записи A;
- использовать общедоступные антиспамовые черные списки
- блокировать спам в электронной почте, проверяя заголовок и текст сообщения.

Список литературы

1 Что такое облачные вычисления? URL: http://azure.microsoft.com/ru-ru/overview/what-is-cloud-computing/ (дата обращения 23.04.2019).

2 Что такое Amazon EC2? Официальная документация AWS URL:https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html/ (дата обращения 23.04.2019).

3 Денисов Д.В. Перспективы развития облачных вычислений. – М.: Издательский дом Университета "Синергия", 2009

4 Серверы. Почтовые серверы URL://https://softportal.com/dlcategory-373-1-0-0-0.html (дата обращения 23.04.2019).

5 Apache и Nginx. Подробный обзор URl:https://hostinger.ru/rukovodsta/web-server-nginx-apace (дата обращения 23.04.2019).

6 Полный синтаксис DKIM, DMARC и SPF URL:https//habr.ru/post/343128 (дата обращения 23.04.2019).

7 Крис Б. Безопасность Linux сервера. Атака и защита. – Уэльс:John Wiley and Sons, Inc:2016.

8 Фленов М. Linux глазами хакера. 5-е издание. – СПб.: БХВ-Петербург, 2019

9 Дэйв В. Защита от хакеров Web-приложений. – М.:Айти , ДМК Пресс:2018.

10 Защита WEB-приложения: практические кейсы. URL: https://habr.com/ru/post/325784/ (дата обращения 24.04.2019)

11 Алексеенко К. Web-сервер глазами хакера. – СПб.: БХВ-Петербург, 2019.

12 Романов П.Ю, Лисьев Г.А Программное обеспечение компьютерных сетей и web-серверов. – М.: Инфра-М, 2015.

13 Жуков Ю. Основы WEB-хакинга. Нападение и защита. – Воронеж: 2011.

14 Хакимжанов Т.Е. Расчет аспирационных систем. Дипломное проектирование. Для студентов всех форм обучения всех специальностей. – Алматы: АУЭС, 2014.

15 Бекишева А.И. Методические указания к выполнению экономической части дипломной работы для бакалавров специальности 5В0703 – Информационные системы – Алматы: АУЭС, 2013.

Перечень сокращений

AWS – Amazon Web Services
SaaS – Software as a Service
PaaS – Platform as a Service
IaaS – Infrastructure as a Service
WEB – World Wide Web
URL – Uniform Resource Locator
WAF – Web Application Firewall
IP – Internet Protocol
ИТ – Информационные технологии
ПО – Программное обеспечение
ОС – Операционная система
FTP – File Transfer Protocol
SQL – Structured Query Language
XSS – Cross-site scripting
DDoS – Distributed Denial of Service
SPF – Sender Policy Framework
DKIM – DomainKeys Identified Mail
DMARC – Domain Message Authentication Reporting & Conformance
HTTP – HyperText Transfer Protocol
HTTPS – Hypertext Transfer Protocol Secure
SMTP – The Simple Mail Transfer Protocol
POP – Post Office Protocol
IMAP – Internet Message Access Protocol
CDN – Content Delivery Network