

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ  
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

**Московский институт электроники и математики**

Коняхин Илья Александрович

**РАЗРАБОТКА РЕЛЕВАНТНОГО СЕРВЕРНОГО ПРОГРАММНОГО КОМПЛЕКСА  
ДЛЯ КОРПОРАТИВНОЙ ЭЛЕКТРОННОЙ ПОЧТЫ**

Выпускная квалификационная работа  
студента образовательной программы

«Информационные системы и технологии»

по направлению 09.03.02 «Информационные системы и технологии»

Студент

И.А. Коняхин

\_\_\_\_\_

подпись

Рецензент,  
доцент ДПМ МИЭМ НИУ ВШЭ,  
к. т. н. А.А. Внуков

Руководитель ВКР,  
к. т. н., проф.  
А.Ю. Истратов

Москва 2016 г.

## **Аннотация**

Рассматривается задача проектирования и внедрения серверного программного комплекса для работы с электронной почтой. На примере открытого и свободного ПО продемонстрирована возможность полностью удовлетворить потребности заказчика в современном сервисе по приему и отправке электронных писем. Помимо этого отдельно уделено внимание на миграцию данных пользователей с устаревшей почтовой системы на новый стек ПО.

## **The Development of Relevant Server Software for the Corporate Email**

by

Ilya Konyakhin

## **Abstract**

*We discover the problem of design and implementation of the server software for e-mail. Using free and open source software as example shows the ability to fully satisfy the needs of the customer in a modern service for receiving and sending e-mails. In addition, the attention is paid at user data migration from legacy mail systems to the new software stack.*

Supervisor: Prof. \_\_\_\_\_

## Содержание

Введение .....	4
1. Анализ существующего почтового решения.....	5
1.1 Основные понятия и термины .....	5
1.2 Описание компонентов текущей архитектуры .....	15
1.3 Постановка задачи по модернизации существующей почтовой архитектуры .....	20
2. Проектирование модернизированного почтового решения.....	21
2.1 Анализ нового решения.....	21
2.2 Обоснование выбора компонентов .....	22
3. Разработка и внедрение модернизированного почтового решения.....	24
3.1 Основные этапы внедрения .....	24
3.2 Выводы .....	29
Заключение .....	30
Список используемой литературы .....	31
Приложения .....	32

## **Введение**

Электронная почта – это комплекс технологий, а так же сервис по отправке, получению и обработке электронных сообщений.

Первые наработки в этой области относят к далекому 1965 году. В наши дни, важность электронной почты для компании сложно переоценить. За счет того, что протоколы работы электронной почты открыты, технические спецификации стандартизированы и описываются в RFC документах, существует множество как и открытых, так и закрытых реализаций этих стандартов. Таким образом, встает множество вопросов какие компоненты выбрать, посредством чего заставить их взаимодействовать друг с другом, обеспечив должную прозрачность работы, легкость внедрения и поддержки, масштабируемость и отказоустойчивость. Помимо всех перечисленных выше проблем выбора отдельно встает вопрос обеспечения безопасности передаваемых данных, сохранение репутации доменного имени, от лица которого ведется рассылка. Отправленные письма не должны теряться и попадать в «спам» на стороне получателя. Обратной, более трудоемкой задачи, противодействие рассылаемому спаму извне.

Данная работа является актуальной и практической полезной. Задача по архитектурному планированию, разработке и развертыванию комплекса ПО электронной почты встает перед всяким, кому требуется сделать это на подконтрольных ресурсах.

Рассматриваются этапы по оценке текущего почтового решения на примере ФГБУ «НЦССХ имени А.Н. Бакулева» Министерства здравоохранения РФ, проектированию обновленного стека ПО для работы с почтой, а так же последующее его внедрение «в бой» с миграцией на новое решение накопленных пользователями электронных писем. На примере стека свободного ПО будет продемонстрировано, как удовлетворить потребности заказчика в релевантном почтовом решении.

# 1. Анализ существующего почтового решения

## 1.1 Основные понятия и термины

В терминологии электронной почты используются следующие компоненты и протоколы:

- MTA (англ. *Mail Transfer Agent*) — отвечает за получение и отправку писем другим компьютерам. Через сеть взаимодействует с другими MTA посредством протокола SMTP. Соответственно выступают либо в роли клиента, либо в роли сервера в зависимости от направления передачи письма. Примеры ПО: Postfix, Exim, Sendmail.
- MDA (англ. *Mail Delivery Agent*) — отвечает за доставку почты конечному пользователю. Позволяет писать правила фильтрации поступающих писем посредством специального языка Sieve. MDA как правило вызывается по запросу от MTA. Примеры ПО: Maildrop, Procmal, Dovecot.
- MUA (англ. *Mail user agent*). Под MUA понимается почтовый клиент, с которым непосредственно работает пользователь и выполняет операции по чтению, написанию, отправке, поиску и архивированию сообщений. Может быть выполнен в виде толстого, тонкого, либо мобильного приложения. MUA работает по протоколам SMTP, IMAP, POP3. Примеры ПО: Mozilla Thunderbird, MS Outlook, Roundcube web-client.
- MRA (англ. *Mail retrieve agent*) — специальный почтовый агент, забирающий почту с другого сервера. Как правило, для этих целей используется POP3. После забора письмо передается MDA. Примеры ПО: Fetchmail, Getmail.
- Форматы хранения писем: mailbox и maildir. Первый является одиночным текстовым файлом, который хранит всю почту пользователя, разделяя сообщения пустой строкой. Для выполнения действий с почтой требует монопольной блокировки на файл. Maildir же копирует модель файловой системы, где каждое сообщение — отдельный файл, а папка в почте пользователя — каталог на файловой системе. Таким образом, для формата maildir не требуется глобальная блокировки всей почты.

- SMTP (англ. *Simple mail transfer protocol*) — основной протокол для взаимодействия почтовых серверов друг с другом. Важным требованием для корректной работы SMTP является правильная настройка DNS-серверов.

- LMTP (англ. *Local Mail Transfer Protocol*) — протокол для передачи почты, предназначенный в первую очередь для локальной пересылки, в глобальной сети не используется. Обычно служит для связки различных почтовых модулей друг с другом, например, проверки на спам и вирусы.

- IMAP (англ. *Internet Message Access Protocol*) — протокол для доступа к электронной почте со стороны MUA. Основное отличие от POP3 – письма остаются на стороне сервера, что имеет ряд преимуществ для конечного пользователя: доступ со множества устройств одновременно, надежность хранения данных, так как локальный ПК зачастую менее надежен чем серверная инфраструктура.

- POP3 (англ. *Post Office Protocol Version 3*) — по функциональному назначению аналогичен IMAP. Принципиальное отличие – скачивает почту на локальную машину пользователя.

- LDAP (англ. *Lightweight Directory Access Protocol*) — протокол применяется для доступа к данным, хранящимся в каталоге, выполнения операций поиска и обновления данных. Обычно применяется там, где операции чтения превалируют над операциями записи.

- Active Directory – проприетарная служба каталогов от компании Microsoft, реализующая LDAP и дополнительные расширения.

- FQDN (англ. *Fully qualified domain name*) — полное имя конкретного компьютера или хоста в Интернете.

- DNS/rDNS (англ. *reverse/Domain Name System*) — распределенная компьютерная система для получения данных о доменных именах и IP-адресах.

Для того, чтобы почтовый сервер мог корректно отправлять и получать письма, требуется прописать следующие DNS записи:

- Запись типа A – определяет соответствие между FQDN и IP-адресом,
- Запись типа PTR – определяет обратное соответствие между IP-адресом и FQDN,
- Запись типа MX – определяет какой хост имеет право принимать почту для данного домена,
- Запись типа TXT – определяет настройки домена для относительно новых стандартов SPF, DKIM, DMARC, разработанных с целью повышения безопасности электронной переписки, противодействия спаму, фишингу и уменьшению паразитного трафика в сети.
- SPF (англ. *Sender Policy Framework*) – стандарт и механизм на базе DNS, позволяющий получателю письма проверить нет ли подлога доменного имени в присланном письме, и верно ли, что письмо действительно пришло с того домена, который указан в заголовке «from:». Когда письмо приходит на МТА получателя, МТА через внешний модуль делает запрос к DNS на наличие TXT записи для домена, с которого якобы отправлена почта. Если у этого домена обнаруживается соответствующая TXT запись, где перечислены все диапазоны IP-адресов и все имена хостов, с которых данный домен может отправлять почту, то происходит сличение IP-адреса из служебного заголовка письма с этим диапазоном. Если сличение успешное, и письмо действительно отправлено с сервера, который входит в данный диапазон, то такое письмо проходит проверку и принимается получателем. Если же нет, то действует одно из трех правил: принять, принять как подозрительное, и отвергнуть. Правила по умолчанию задаются в той же TXT записи и регламентируются доменом отправителя.
- DKIM (англ. *DomainKeys Identified Mail*) – метод аутентификации, то есть проверки на подлинность, позволяющий получателю проверить легитимность полученного письма, подобно тому как это делается в стандарте, описывающим SPF. Принципиальная разница в том, что выполняется не просто сличение IP-адреса, а проверка электронной цифровой подписи (ЭЦП) полученного письма посредством

технологии асимметричного шифрования. Получатель проверяет, если ли ЭЦП у письма, которая создается через модули, работающие в связке с MTA на стороне отправителя. Если ЭЦП есть, то делается запрос к соответствующей TXT DNS-записи домена отправителя, где считывается открытый ключ и на его базе проверяется целостность подписи. В результате проверки получатель видит, прошло ли письмо тест на совместимость с DKIM или нет.

- DMARC (англ. *Domain-based Message Authentication, Reporting and Conformance*, RFC 7489) – стандарт, принятый в 2015 году, разработанный с целью обеспечить прозрачный механизм приема и отклонения писем, если для доменов прописаны соответствующие правила, а так же позволяющий получать отчеты о письмах, при не прохождении проверки. DMARC использует для своего функционирования DNS записи типа TXT. DMARC устраняет самую серьезную проблему электронной почты – отсутствие проверки адреса отправителя. SPF не защищает адрес отправителя, а работает только с отправителем конверта (envelope from:). DKIM не реализует механизма правил, что делать с письмами, если они не прошли DKIM проверку и не защищает поле отправителя (from:).

В качестве протоколов авторизации DMARC использует протоколы SPF и DKIM. Если полученное письмо проходит хотя бы одну из двух проверок, то к нему применяется правило DMARC – отклонить прием письма, отправить в спам, принять как есть. Правила обработки письма аналогично SPF регламентируются администратором домена. Преимущество DMARC относительно всех остальных, так и не получивших широкого распространения, политик: простая настройка – используются понятные правила на базе DNS, умеет авторизовываться по SPF и по DKIM, отсутствие патентованных технологий, широкая поддержка ведущими сервисами электронной почты в Интернете.

- Спам – нежелательная электронная почта, как правило рекламного характера



- Фишинг – мошенническая почта, вводящая получателя в заблуждение, с целью получения неправомерного доступа к личным данным.

- Greylisting – метод борьбы со спамом, на основе «серых списков» IP-адресов. Суть заключается в задержке приема письма от клиента с отправкой ответа от МТА о невозможности принять почту. Правильно настроенные клиенты попробуют отправить письмо еще раз через некоторый интервал времени. Если это произойдет, то МТА примет письмо. Методика работает, потому что в глобальной сети спам составляет до 95% трафика, спамерам не выгодно копить очередь отказов в приеме и пробовать открыть соединение еще раз.

- Honeypot (англ. «горшочек с мёдом») – специальный незащищенный ресурс в глобальной сети, задача которого привлечь на себя внимание злоумышленников. В контексте почты honeypot представляет собой открытый релей, который якобы принимает запросы на доставку писем от кого угодно без какой-либо авторизации. Honeypot не раскрывается легитимным пользователям и полезной нагрузки на нем быть не должно априори. Соответственно все письма, которые доходят по этого сервера, являются мусорными. Естественно эти письма не пересылаются дальше по сети, а анализируются для составления черных списков клиентов.

- DNSBL (англ. *DNS-based Blackhole List*) – метод борьбы со спамом на основе черных списков IP-адресов и хостнеймов, которые рассылают спам. Списки формируются путем анализа активности на серверах-приманках – honeypot.

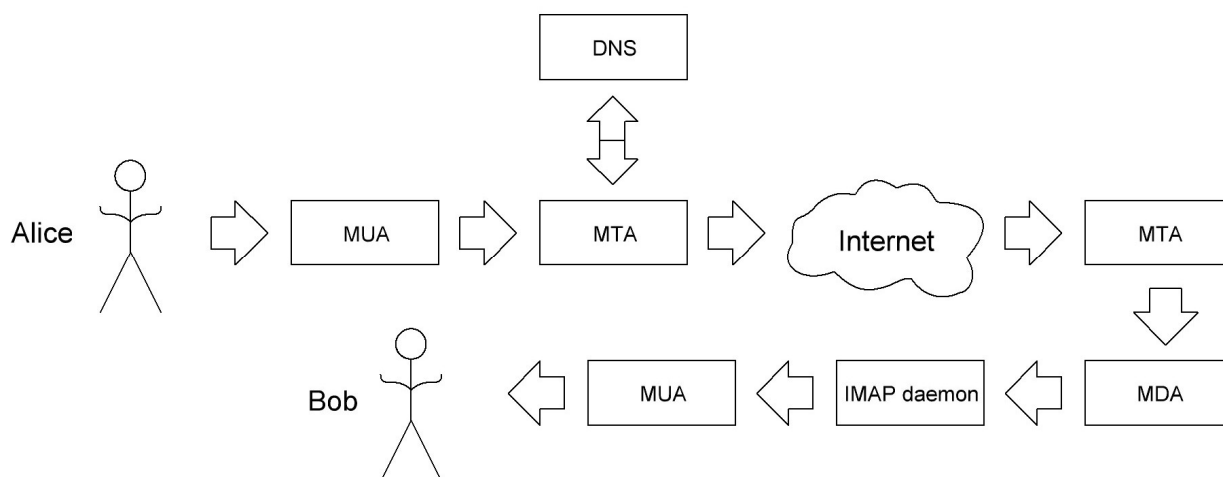
- DoS (англ. *Denial-of-service*) – отказ в обслуживании. Под этим понимается невозможность сервиса корректно выполнять свои функции из-за частичной, либо полной недоступности.

Помимо терминологии стоит понимать общий принцип работы электронной почты. Структурная схема процесса отправки и приема отображена на рис. 1.

По шагам этот процесс происходит так:

- Алиса хочет отправить письмо Бобу.

- Алиса отправляет письмо на test@example.com, письмо передается из её MUA на её MTA.
- MTA обрабатывает очередь всех полученных писем.
- MTA запрашивает через DNS MX-запись сервера, который обслуживает почту для домена example.com.
- MTA устанавливает связь через Интернет с MTA, который обслуживает домен example.com.
- MTA получателя выполняет все необходимые проверки отправителя, и если проверки прошли успешно, принимает письмо. В противном случае либо отказывает в приеме на время с просьбой повторить попытку позднее, либо отказывает вовсе.
- Если Алиса ошиблась с адресом Боба и написала на несуществующий ящик, то от её MTA ей придёт отбойник, что сообщение не может быть доставлено.
- MTA получателя, Боба, передает письмо MDA - агенту доставки.
- Агент доставки может выполнить операции по сортировке письма через правила на языке Sieve, которые настраиваются каждым пользователем индивидуально. MDA так же анализирует заголовки, которые добавил MTA, например, является ли письмо спамом или нет. Если да, то MDA перемещает письмо в директорию для спам писем.
- Боб заходит в свой почтовый агент - MUA, который инициирует запросы к IMAP-серверу. IMAP-сервер считывает письмо, хранящееся в почтовом ящике и передает его по сети в почтовый клиент Боба.
- С этого момента Боб может прочесть письмо.



**Рис. 1.** Структурная схема отправки и приема почты

На рис. 1, две стрелки до и после облачка Internet отображают собой SMTP-сессию, которая является ключевым моментом взаимодействия почтовых серверов друг с другом. Именно в этой сессии определяется, будет ли принято письмо, сможет ли оно быть доставлено получателю, можно ли доверять отправителю, либо стоит применить к нему одну из ограничивающих политик. Понимание работы SMTP-сессии – ключевой навык для правильной настройки работы почты. На этом этапе сетевого взаимодействия двух почтовых серверов отсеивание спама и иной нежелательной почты является наименее ресурсоёмкой операцией. Это первый эшелон защиты почтовой инфраструктуры, отсеивающий до 95% всех запросов на отправку письма.

На примере домена bakulev.ru показано, как устанавливается SMTP-сессия. Для начала надо узнать какой хост обслуживает домен. Для этого клиент обращается к DNS с таким запросом:

*dig -tMX bakulev.ru*

И получает ответ:

*;; ANSWER SECTION:*

*bakulev.ru. 900 IN MX 10 mail-gateway.bakulev.ru.*

Далее FQDN *mail-gateway.bakulev.ru.* конвертируется в IP-адрес таким запросом:  
*dig -ta mail-gateway.bakulev.ru.*

В качестве ответа имеем IP-адрес 46.46.152.210:

*:: ANSWER SECTION:*

*mail-gateway.bakulev.ru. 30 IN A 46.46.152.210*

Затем устанавливается соединение с IP-адресом 46.46.152.210 и стандартным для SMTP-протокола портом 25. В самом простом формате, без дополнительных проверок со стороны МТА, сессия описана ниже.

1. Открывается TCP-соединение с хостом 46.46.152.210 и портом 25:

*telnet 46.46.152.210 25*

*Trying 46.46.152.210...*

*Connected to 46.46.152.210.*

*Escape character is '^]'.*

2. МТА получателя приветствует клиента:

*220 mail-gateway ESMTP Postfix (Debian/GNU)*

3. Клиент приветствует МТА:

*helo example.com*

МТА устраивает такое приветствие и он его принимает:

*250 mail-gateway*

4. Клиент без всякой авторизации пишет от чьего адреса хочет отправить письмо:

*mail from: <me@example.com>*

МТА разрешает принять почту от адреса *me@example.com* и продолжать SMTP-сессию:

*250 2.1.0 Ok*

5. Клиент пишет кто является получателем письма:

*rcpt to: <iakonyakhin@bakulev.ru>*

МТА согласен принять письмо в пользу получателя *iakonyakhin@bakulev.ru*:

250 2.1.5 Ok

6. Клиент инициализирует передачу самого тела письма:

*data*

354 End data with <CR><LF>.<CR><LF>

*From: Ilya Konyakhin*

*To: Ilya A Konyakhin*

*Subject: test from Internet*

*Hi, just test. Ignore it.*

.

7. МТА получателя подтверждает, что принял письмо и присвоил ему уникальный идентификатор в своей очереди на доставку – DA80E24A0A.

250 Ok: *queued as DA80E24A0A*

8. Клиент завершает соединение:

*quit*

221 Bye

Как показано выше SMTP протокол не зря считают простым. Тем не менее его простота является его же слабостью. Любой человек в сети, может отправить письмо совершенно от любого адреса, породив тем самым две основные почтовые проблемы: нежелательная корреспонденция – спам и мошеннические письма.

Для защиты SMTP от подобных изъёнов применяется связка технологий и стандартов в виде SPF, DKIM, DMARC. Помимо этого сами МТА умеют выполнять базовые проверки клиентов, желающих отослать письмо. МТА реализуют множество техник, для уменьшения количества нежелательной почты. Проверки SMTP-сессии выполняются на уровне клиента, приветствия HELO/EHLO, заголовка отправителя конверта, заголовка получателя конверта.

Самые ресурсоёмкие проверки выполняются на этапе передачи тела письма. Поэтому следует вовсе отказаться от них и переложить проверку на сторону

специализированных антиспам решений, либо минимизировать проверки на стороне МТА.

На рис. 2 приведено соответствие этапов SMTP-сессии и типов проверок, которые проводит МТА.

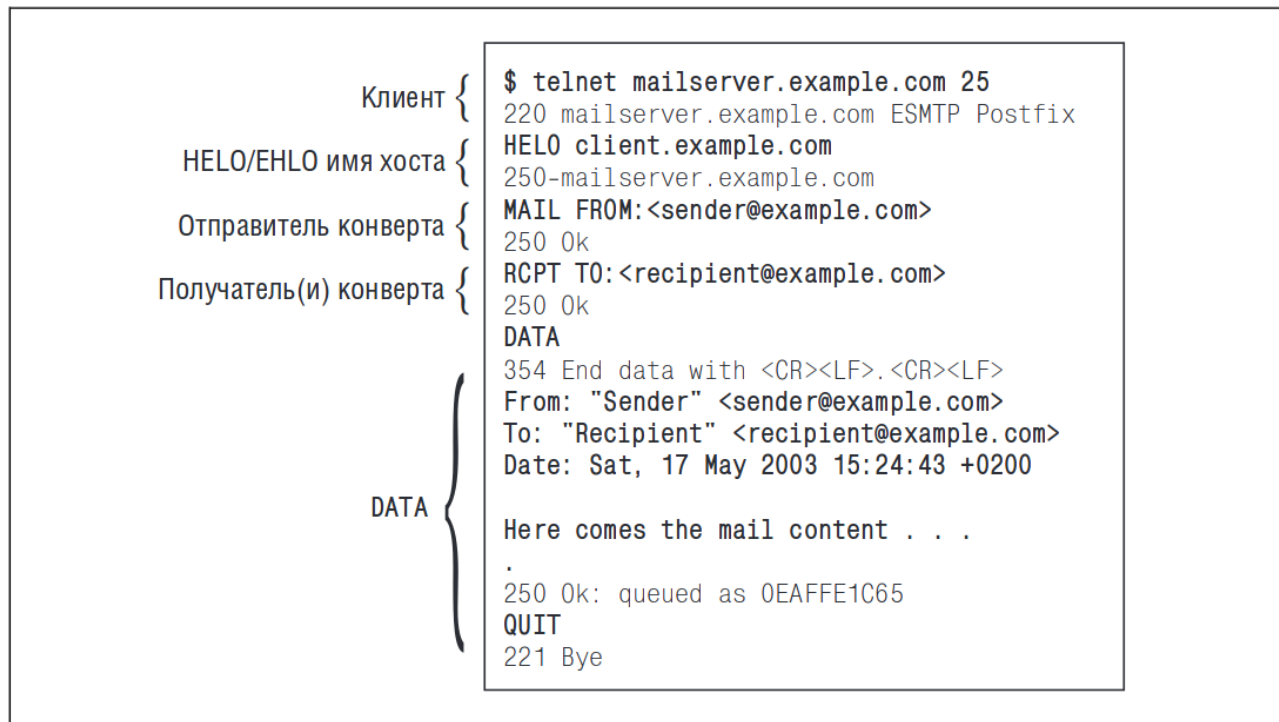


Рис. 2. Этапы SMTP-сессии и клиентского ввода

## 1.2 Описание компонентов текущей архитектуры

В приложении 1 приведена схема текущей почтовой конфигурации.

Конфигурация состоит из следующих элементов:

- Шлюз mail-gateway.bakulev.ru, который принимает входящие соединения от клиентов из Интернета, обрабатывает SMTP-сессии в соответствии с политиками приема писем, осуществляет проверки по заголовкам, приведенным на рис. 2. Физически шлюз является виртуальной машиной под управлением дистрибутива Debian GNU/Linux, MTA Postfix, модуля к Postfix для проверки SPF.
- Статистический спам-фильтр Dspam осуществляет проверку входящих писем на спам путем присваивания каждому принятому письму заголовка X-DSPAM-Result. Результат проверки может быть Innocent, White-list, Spam. После добавления заголовка письмо передается далее на почтовый сервер mail.bakulev.ru. Dspam фактически запущен на той же виртуальной машине, что и Postfix. Общение Dspam и Postfix осуществляется через протокол LMTP. Dspam для эффективной работы должен постоянно обучаться на базе заранее отсортированных писем по признаку спам или легитимное письмо. Если такого обучения не проводить перманентно, база данных спам-фильтра быстро становится неактуальной и не дает желаемого уровня фильтрации. Для доставки до Dspam обучающего контента используется MRA getmail, который скачивает часть отсортированной почты пользователей.
- Почтовый сервер mail.bakulev.ru является ключевым элементом архитектуры. Этот сервер хранит всю почту пользователей, осуществляет авторизацию сотрудников в каталоге Active Directory, отправку писем в локальную и глобальную сеть. Выполняет сортировку писем на основании служебных заголовков, например, наличие X-DSPAM-Result=\*\*\*SPAM\*\*\* в письме будет обработано как спам и соответственно попадет пользователю в директорию для спама. Почтовый сервер работает на основе свободного ПО Hmail server, который выступает в роли MTA, MDA,

MRA и IMAP/POP3-демона. Таким образом, списки рассылок, алиасы, подписание писем ЭЦП реализуются силами Hmail server. Для хранения всех настроек и путей к письмам используется БД MySQL. Физически mail.bakulev.ru представляет собой виртуальную машину на базе ОС Microsoft Windows.

- Веб-интерфейс <https://mail.bakulev.ru/> позволяет сотрудникам получать доступ к почте и адресной книге на базе Active Directory через браузер. Адрес может быть открыт и из глобальной сети, что позволяет не настраивать почтовый клиент, MUA, дополнительно каждому сотруднику. Реализован на базе ПО Roundcube, написанного на PHP+JS, использует веб-сервер Apache, БД MySQL. Физически запущен на виртуальной машине rc.bakulev.ru под управлением Debian GNU/Linux.

Можно заметить, что веб-интерфейс и почтовый сервер имеют одинаковое доменное имя, но запущены на разных виртуальных машинах с разными IP-адресами. Таким образом, если сотрудник заходит по ссылке <https://mail.bakulev.ru/>, он попадает в начале на веб-сервер Nginx, запущенный на mail.bakulev.ru. Nginx выступает в роли проксирующего веб-сервера и отправляет все запросы к веб-серверу Apache, запущенный на виртуальной машине rc.bakulev.ru.

Описанная выше архитектура имеет несколько недостатков, которые не позволяют в полной мере пользоваться всеми преимуществами электронной почты. К этим недостаткам можно отнести отсутствие комплексного спам-фильтра, который умеет не просто определять письмо как спам/неспам, а давать качественную оценку письму, выраженную в баллах, основываясь на множестве факторов. Существует ряд методик противодействия спаму и фишингу: статистический анализ тела письма, проверка IP-адреса клиента на присутствие в черном списке спамеров, корректность приветствия в рамках SMTP-сессии, проверка существования домена отправителя и его почтового ящика, наличие A, MX, PTR записей у домена отправителя, проверка существования адреса и домена получателя. Каждой из этих проверок в многофакторном спам-фильтре можно назначить определенный вес. Часть проверок,



могут иметь отрицательные баллы. Например, наличие у доменной зоны корректных правил записей SPF, DKIM дает отрицательные баллы и позитивно влияет на репутацию домена. Конечно, ничто не мешает спамерам настроить эти записи для своих подложных доменов, но это требует сил, времени и квалификации, что в итоге ведет к удорожанию услуг спамеров для заказчика и делает почтовый спам менее привлекательным для проведения рекламных акций.

После всех проверок письмо получает конечный балл, который является рациональным числом. Чем он больше, тем больше вероятности, что письмо – спам. Далее в спам-фильтре задаются пороги, какую применить политику к письму, в зависимости от того, сколько баллов это письмо набрало.

К сожалению, текущее решение в виде Dspam работает только по статистическому принципу, о многофакторности нет и речи. Нет технической возможности гибко настроить пороги, нет полноценного грейлистинга, он строго реализуется только на стороне модуля к MTA Postfix и дает посредственные результаты, нельзя избирательно включить проверку клиента через DNSBL. Аналогично грейлистингу проверку через DNSBL можно включить на стороне Postfix, но это дает неудовлетворительный результат, так как нет того синергетического эффекта от множества DNSBL, который можно получить от многофакторного спам-фильтра. Включения неоднозначных проверок DNSBL, применение грейлистинга на базе Postfix, либо модулей к нему, приводит к недоставке части легитимной корреспонденции или ошибке второго рода, что всегда хуже ошибок первого рода, когда спам считается легитимной корреспонденцией.

Помимо этого обучение Dspam производится через два технических почтовых ящика `spam@bakulev.ru` и `notspam@bakulev.ru`. В первый почтовый ящик приходит почта на так называемый catch-all почтовый адрес, то есть туда приходят все письма, отправленные на несуществующие ящики. В 99% случаев это спам. Это позволяет иметь источник актуального спама, пригодного для обучения фильтра. Но есть и

обратная сторона. Если внешний клиент со стороны неправильно записал почтовый ящик сотрудника, либо просто опечатался, то, во-первых, клиенту не придет отбойник о недоставке сообщения, так как сообщение было принято и попало в `spam@bakulev.ru`, а с другой стороны эти ошибочные письма портят обучение и статистическую модель, привнося в нее лишние искажения. В свою очередь ящик `notspam@bakulev.ru` дает базу для обучения легитимным письмам. Письма на ящик `notspam@bakulev.ru` не попадают в автоматическом режиме, а приходят от пользователей, которые заметили ошибку второго рода, то есть пометка как спам полезного письма, и не поленились перенаправить письмо на ящик `notspam@bakulev.ru`.

Как показала практика, обучение по двум ящикам недостаточный для полноценной фильтрации метод и не может отражать интересы всех сотрудников. Например, для отдела продаж рассылка о проведении выставки-продажи – это легитимное письмо, а для технического отдела – спам. Dspam изначально был настроен на работу с общей статистической базой. Таким образом, еще один очевидный минус Dspam в текущей конфигурации – это отсутствие полноценной обратной связи от пользователей. Сотрудник переносит письмо из входящих в директорию спам своего ящика, но спам-фильтр этому не обучается, так как нет полной интеграции между шлюзом `mail-gateway.bakulev.ru` и между почтовым сервером `mail.bakulev.ru`.

Все эти проблемы снижают качество фильтрации и общую удовлетворенность пользователей работой почты, но, по крайней мере, не приводят к другому типу проблем – отказ в обслуживании. В текущей конфигурации время от времени возникал DoS из-за нехватки памяти на стороне БД MySQL, которая хранила токены Dspam на `mail-gateway.bakulev.ru`. Как видно по схеме из приложения 1, Postfix отправляет письма на анализ в Dspam через локальный интерфейс `127.0.0.1` по протоколу LMTP. Когда БД MySQL падала из-за нехватки памяти, Dspam тоже падал и не мог фильтровать письма. В очереди сообщений Postfix накапливалось множество сообщений, которые ждали обработки спам-фильтром. Без этой обработки сообщения не могли быть доставлены до

mail.bakulev.ru. Это приводило к жалобам от пользователей.

Итого, недостатки использования DSPAM таковы:

- Только статистический анализ,
- Невозможность компромиссного использования greylisting и DNSBL,
- Нет полноценной обратной связи от пользователей,
- Легитимные письма с ошибками в адресе не доходят до получателя и отправитель об этом не уведомлен,
- Случаются ситуации, которые приводят к DoS. Входящий поток почты намертво прекращается,
- Dspam перестал развиваться, последний коммит в дерево исходного кода был в начале 2015 года, поддержка современных дистрибутивов прекращена.

Помимо Dspam, слабым звеном является сам Hmail server. Его недостатки следующие:

- Нет полной интеграции с антиспамом, сотрудники не могут обучать фильтр самостоятельно.
- Hmail server использует для своей работы БД, которая создает дополнительную нагрузку на виртуальную машину и со временем требует выноса СУБД на отдельный хост. При наличии AD, использование БД является бессмыслицей, порождающей лишнее звено отказа.
- На хосте mail.bakulev.ru установлен проксирующий веб-сервер Nginx, он обрабатывает запросы к <https://mail.bakulev.ru> из глобальной сети. Если в Nginx, либо модуле, который использует Nginx, появляется уязвимость, то приходится обновлять Nginx вручную. Mail.bakulev.ru работает под управлением ОС Windows, поэтому об удобных репозиториях с обновлениями ПО говорить не приходится. Таким образом, возрастают расходы на обслуживание ОС и третьих программ.
- ОС Windows требует покупки лицензии, что не оправдано там, где можно обойтись free & open source решениями.

### **1.3 Постановка задачи по модернизации существующей почтовой архитектуры**

Была поставлена задача улучшить работу спам-фильтра, добавить обратную связь от пользователей, убрать лишние звенья в цепочке по обработке почты и упростить администрирование, уйти от использования ОС Windows и Hmail Server.

В качестве основной цели стояло уменьшение негатива от конечных пользователей на нестабильность работы почты, неэффективность работы спам-фильтра. Дополнительная цель заключалась в высвобождении лицензии на ОС Windows и сосредоточении всего стека для работы с электронной почтой на ОС Debian GNU/Linux. Это позволит добиться гомогенности почты под управлением одной версии ОС, что открывает возможности удобного администрирования и эксплуатации.

Помимо описанного выше комплекса ПО, предназначенного для обработки почты сотрудников, в организации используется несколько сторонних площадок для рассылки почты. Требовалось внедрить SPF, DKIM, DMARC для основного домена и технических поддоменов для улучшения репутации всех отправляемых организацией писем, а также мониторинга спам-активности внутри сети.

Таким образом термин «релевантность», вынесенный в заголовок работы, предполагает соответствие критериям:

- Безопасности,
- Интегрированности компонентов,
- Удобства администрирования и поддержки,
- Низких накладных расходов на инфраструктуру.

## **2. Проектирование модернизированного почтового решения**

### **2.1 Анализ нового решения.**

В приложении 2 приведена схема переработанной почтовой конфигурации. Элементы в жирной рамке либо претерпели изменения, либо были добавлены впервые.

Как видно из схемы, Dspam полностью заменяется на новое решение и выводится из эксплуатации. На его место приходит спам-фильтр Rspamd и множество модулей, которые позволяют убрать лишние проверки с МТА Postfix. Таким образом конфигурационный файл Postfix становится еще проще и понятнее для изучения. МТА занимается только своей непосредственной работой, не создает дополнительных соединений со сторонними модулями, а передает все сообщения через единственный интерфейс Rmilter. Задача Rmilter – прогнать каждое письмо по цепочке проверок и выдать вердикт в баллах. Далее Rspamd принимает решение что делать с письмом – принять, отправить на грейлистинг, принять и пометить как спам, либо отвергнуть.

Rmilter позволил реализовать проверку на вирусы силами эвристического движка ClamAV и базами сигнатур к нему. Ранее проверка на вирусы не проводилась, на МТА стояли настройки, запрещающие передачу исполняемых файлов в приложениях к письму.

Hmail server заменяется связкой МТА Postfix и IMAP-сервера Dovecot. Данная связка является практически классикой почтовых конфигураций в юниксоподобной среде. Решается проблема с обучением и обратной связью в силу того, что пользовательские данные хранятся теперь в стандартной unix-like директории, могут быть просто идентифицированы как спам и отправлены на дообучение спам-фильтра. Уменьшаются и системные требования, отдельный хост под СУБД теперь не нужен.

## 2.2 Обоснование выбора компонентов

Rspamd – это современный спам-фильтр, имеет множество преимуществ относительно Dspam:

- Интеграция с МТА происходит через стандартный интерфейс коммуникации Rmilter, что позволяет исключить проблему DoS и задает политику по умолчанию «принимать всё», если по каким-то причинам спам-фильтр будет недоступен,
- Rmilter позволяет подключать любые совместимые проверки, например, на вирусы, при этом не блокировать работу почтового шлюза в случае DoS,
- Rspamd присваивает письмам балл степени спамности, что позволяет гибко настраивать пороги срабатывания,
- Rspamd умеет работать с грейстингом и гибко его включать для сомнительных писем. В конфигурации с Dspam грейстинг был возможен только глобально.
- Имеет плагины для проверок по политикам SPF, DKIM, DMARC, DNSBL, SURBL. Умеет проверять репутацию по IP, подсети, стране,
- Выполняет фильтрацию контента по алгоритму поиска заданных параметров в письмах. Выполняет проверку по нечеткому хешу – поиск похожих писем и сравнение с актуальной общедоступной базой спама,
- Умеет статистический анализ подобно Dspam, хранит данные в СУБД sqlite3,
- Позволяет писать модули и расширения на скриптовом языке lua,
- Имеет веб-интерфейс для базовой конфигурации, обучения, проверки статуса.

Rspamd написан на чистом C, имеет событийную архитектуру, что позволяет избежать об блокировок и выполнять параллельные проверки множества писем, тем самым обеспечив высокую производительность, пропускную способность и

нетребовательность к системным ресурсам.

Помимо всех технических преимуществ, Rspamd имеет открытый код и распространяется свободно.

Связка Postfix+Dovecot хорошо себя зарекомендовала на множестве инсталляций в Интернете. Данное ПО имеет свободную лицензию и открыто распространяется. Написано с упором на безопасность и производительность. У обоих продуктов не было zero-day уязвимостей в стандартной конфигурации за последние годы, что позволяет говорить о должной безопасности приложений, с учетом их широкого распространения в глобальной сети. Около 32% всех почтовых серверов в сети используют Postfix как минимум в качестве входящего шлюза.

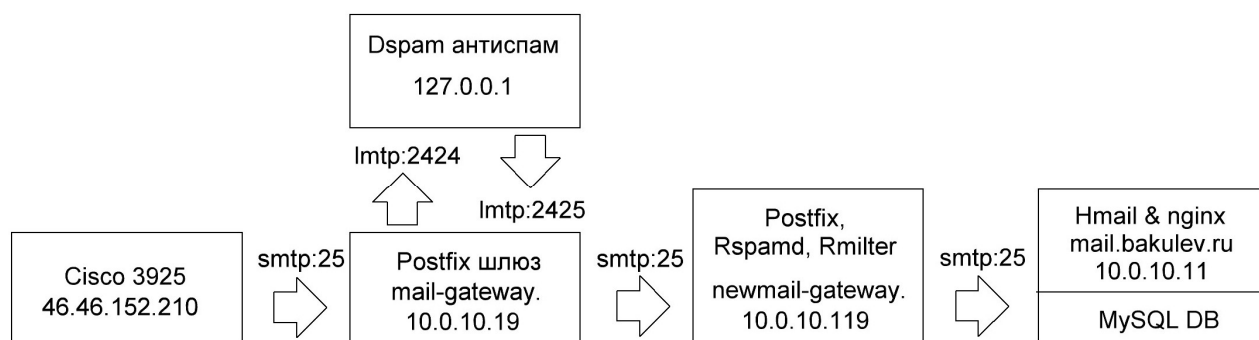
В пункте 1.2 данной работы были описаны основные проблемы, которые не позволяли предоставлять качественную услугу электронной почты.

Замена компонентов из пункта 1.2 на компоненты из пункта 2.1 является достойным решением архитектурных проблем и позволяет убрать все недостатки текущего почтового стека.

### 3. Разработка и внедрение модернизированного почтового решения

#### 3.1 Основные этапы внедрения

Внедрение началось с развертывания спам-фильтра Rspamd и MTA Postfix в виде отдельной промежуточной виртуальной машины, рис. 3.



*Рис. 3. Развертывание нового шлюза в тестовом формате*

В таком режиме надежность конфигурации принципиально не ухудшалась и при этом появилась возможность оценить все функции нового решения «на бою». Одно из многих достоинств Rspamd состоит в том, что он практически сразу работает «из коробки», какого-то специального конфигурирования не потребовалось.

Единственное, что пришлось исправить на стороне Rspamd, это завысить пороги срабатывания, настроенные по умолчанию. По той причине, что почта на новый шлюз приходила не напрямую из Интернета, а от старого шлюза, это приводило к тому, что часть проверок работала некорректно. Проверки, определяющие репутацию IP-адреса, а также проверки DNS, автоматически давали максимальный балл спамности письма. Можно было на время тестирования отключить эти проверки, но более эффективно – завысить глобальные пороги, что и было сделано.



Пороги срабатывания хранятся в файле */etc/rspamd/metrics.conf* и имеют значения по умолчанию:

```
actions {  
    reject = 15;           //если письмо набирает 15 баллов, оно отвергается.  
    add_header    = 6;    //если письмо набирает 6 баллов, оно помечается как  
спам.  
    greylist = 4;        //если письмо набирает 4 балла, оно уходит на грейлистинг, и  
                           //если клиент повторит передач, принимается во входящие.  
};
```

Каждая из политик получила по +6 баллов на время тестирования.

После того, как Rspamd в течение недели показал свою жизнеспособность и эффективность, началась миграция проверок со стороны МТА и сторонних модулей со старого шлюза в конфигурацию Rspamd. Из конфига Postfix на шлюзе, смотрящем в Интернет, были убраны проверки SPF, DKIM.

Еще через неделю, старый почтовый шлюз был выключен, конфигурационный файл МТА Postfix полностью перенесен на новый шлюз, пороги срабатывания вернулись к значению по умолчанию. Виртуалка newmail-gateway стала mail-gateway и IP-адрес был сменен с 10.0.10.119 на 10.0.10.19.

Со стороны сетевого оборудования конфигурация осталась прежней.

После месяца мониторинга за ящиком spam@bakulev.ru, было принято решение отказаться от технических обучающих почтовых ящиков в виду того, что туда практически перестали попадать спам сообщения. В основном там оседали легитимные сообщения с ошибками в названии. Catch-all ящик был перенастроен на секретариат, разбирающий такие сообщения.

На рис. 4 показано распределение количества писем с баллами от -16, до 77. Эта статистика была получена после анализа логов работы Rspamd за неделю. Красным пунктиром обозначены пороги срабатывания политик по умолчанию. Зеленым –

выставленные сейчас «на бою».

В приложении 3 представлен основной конфигурационный файл MTA Postfix – */etc/postfix/main.cf*, взятый с нового шлюза.

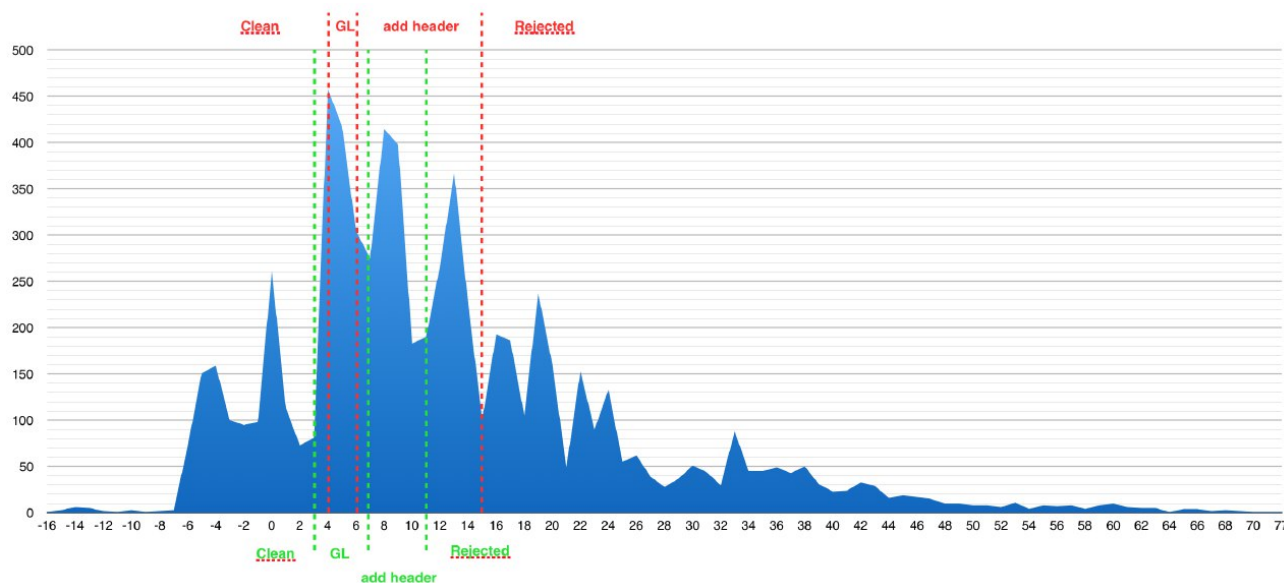


Рис. 4. Распределение полученных писем по баллам и количеству

Далее предстояла задача развернуть связку MTA Postfix + Dovecot, что является достаточно тривиальной задачей в типовых конфигурациях. Наша конфигурация отличается от типовой тем, что настраивается двойная доставка писем сразу на оба сервера. Старый сервер обслуживает пользователей, а новый тестируется.

Newmail.bakulev.ru выступает подобно кеширующему прокси серверу, сохраняя все письма, проходящие через себя и прозрачно перенаправляя их на старый почтовый сервер. Данная возможность реализуется через скрипт *smtpdd.sh*, обрабатывающий очередь и утилиту доставщика почты *msmtp*.

Задача по обеспечению обратной связи от пользователей решается путем монтирования директории с письмами на сторону шлюза, и запуска на нем команд:

```
rspamc -h 10.0.10.19:80 -P password learn_spam /var/mail/%d/u%/spam/*,
```

```
rspamc -h 10.0.10.19:80 -P password learn_ham /var/mail/%d/u%/notspam/*
```

На рис. 5 изображена временная схема параллельной работы двух почтовых

серверов.

В приложении 4 представлена часть параметров конфигурационного файла `master.cf`, отвечающего за двойную доставку.

Дополнительно следует убедиться, что в файле `main.cf` прописан параметр:

`relay_domains = bakulev.ru`

В приложении 5 представлен исходный текст скрипта `smtpdd.sh` для двойной доставки.

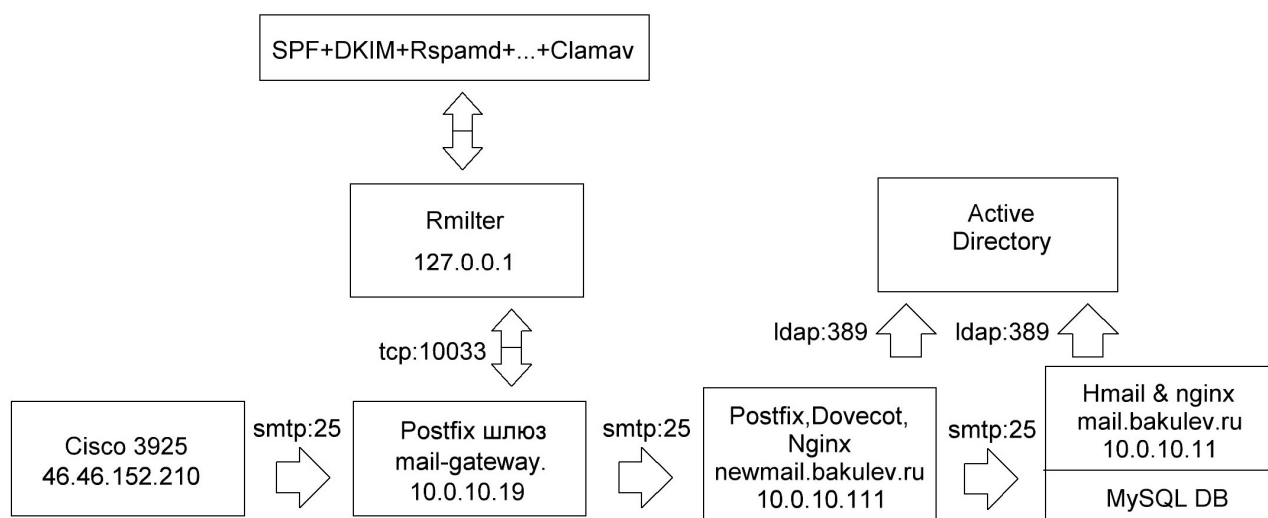


Рис. 5. Временная схема двойной доставки

По состоянию на конец мая 2016 года, новый сервер `newmail` пока не заменил собой старый `mail`. Связано это с тем, что требуется миграция писем пользователей.

Для миграции можно использовать утилиту `imapsync`. Конфигурационных файлов она не требует, достаточно передать все параметры в команде:

```
imapsync --host1 mail.bakulev.ru --user1 $USER --passfile1 /home/mail/secret1 --host2  
newmail.bakulev.ru --user2 $USER --passfile2 /home/mail/secret1
```

В переменной `$USER` можно объявить список все ящиков, которые нужно смигрировать. В ключах `--passfile1` и `--passfile2` передаются пароли от ящиков.

Как говорилось выше, на старом сервере используется AD авторизация, поэтому и

все настройки хранятся в БД. Поэтому для синхронизации требуется развернуть тестовую виртуалку с бекапом БД, выполнить SQL-скрипт:

```
UPDATE hm_accounts
```

```
SET accountisad = '0';
```

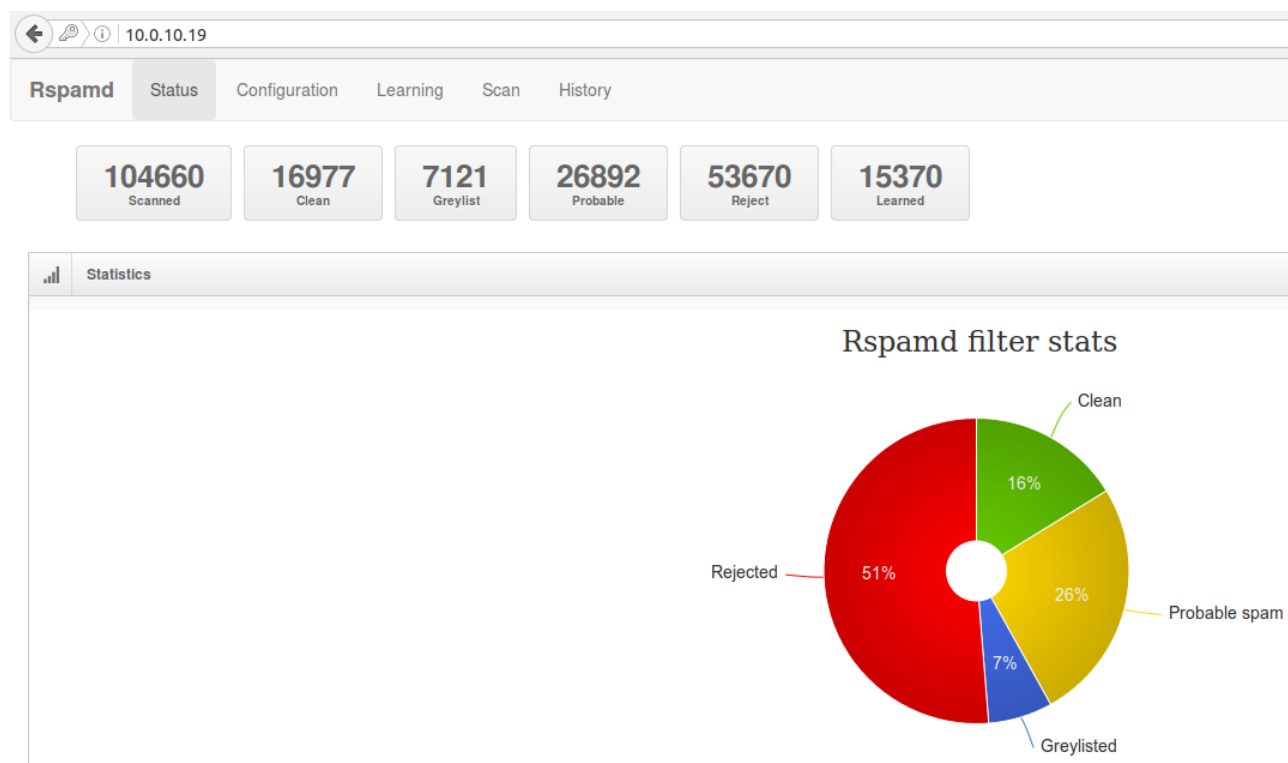
Этот скрипт выключает аутентификацию на стороне службы каталогов, после чего в учетные записи пользователей можно будет зайти по локальному паролю. Так как локальный пароль каждого пользователя не известен, то выполним update поля accountpassword, которое хранит хеш пароля. Сделаем это глобально, теперь все учетные записи для синхронизации доступны под одним паролем.

На текущий момент синхронизация опробована, она работает и позволяет выполнить миграцию.

### 3.2 Выводы

Миграция на новое решение частично завершена. Для полной миграции требуется перенести оставшиеся письма пользователей и запустить новый почтовый сервер в работу. Тем не менее, текущий стек уже использует новые полученные преимущества. Как видно из рис. 6, теперь на втором уровне защиты фильтруется 77% принятой почты. 16% считается легитимной, оставшиеся 7% тяжело проанализировать. В абсолютных числах общий легитимный почтовый трафик составляет ~400-450 писем в неделю. Соответственно новый фильтр дал качественное улучшение фильтрации нежелательной почты в 2.5 раза.

После окончания миграции почты и вывода старого почтового сервера из эксплуатации можно ожидать улучшения статистики за счет полноценной работы обучения и обратной связи.



*Рис. 6. Статистика работы спам-фильтра за 3.5 месяца работы*

## **Заключение**

В результате проведенной работы удалось обеспечить больший уровень доступности почтовой системы. Спам стал фильтроваться более тщательно, используя оценку множества факторов. Появилась возможность тонкой и гибкой настройки политик фильтрации. Для домена и субдоменов прописаны все необходимые записи.

В итоге основная цель работы достигнута – жалобы от пользователей на недоступность и плохую работу спам-фильтра прекратились.

Тем не менее, работу полностью выполненной считать нельзя по причине незавершенной миграции данных. С технической стороны никаких ограничений на миграцию нет, все компоненты настроены и внедрены.

## Список используемой литературы

### Книги:

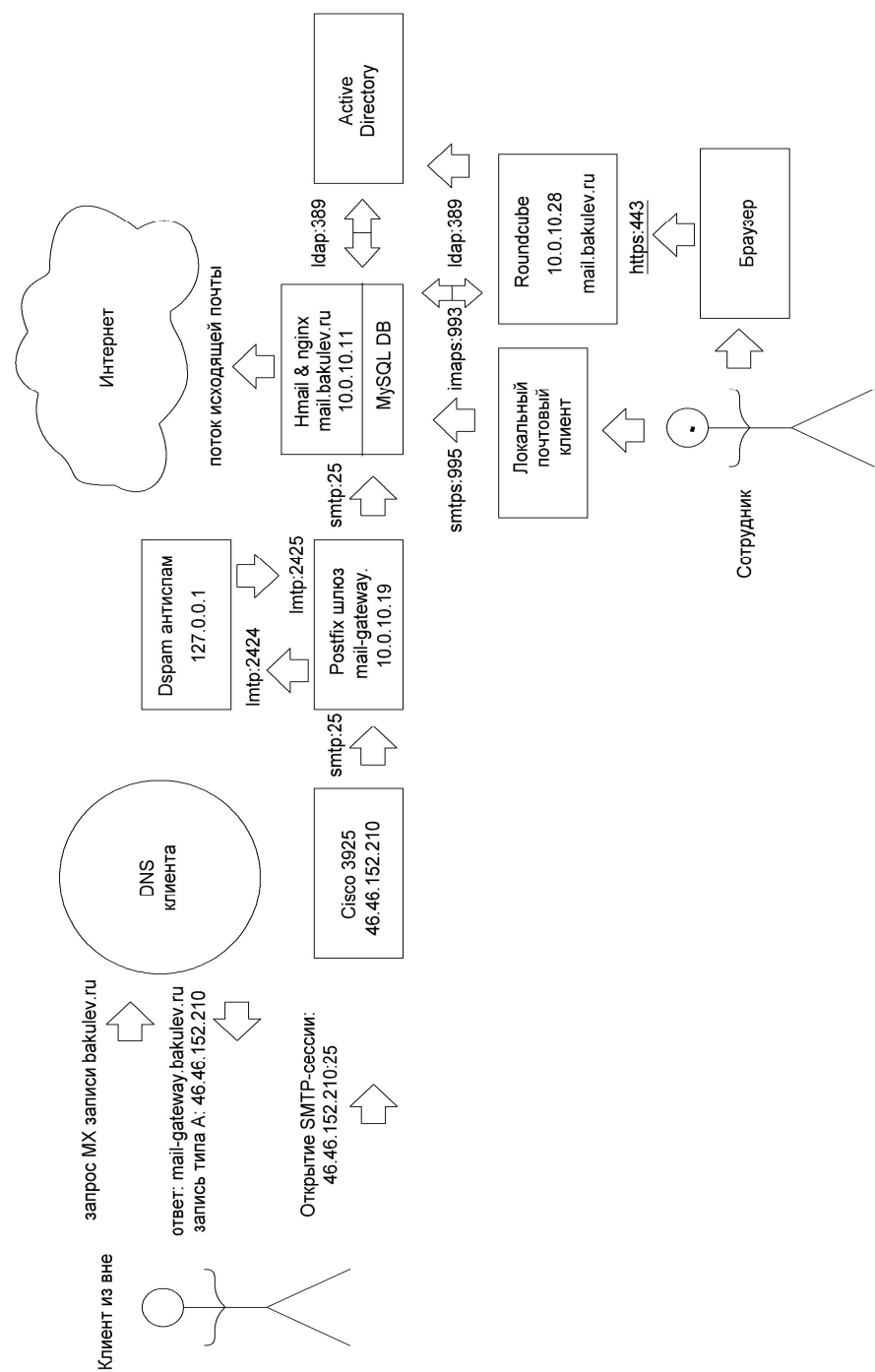
1. Гильдебрандт Р., Кеттер П. Postfix. Подробное руководство. – Пер. с англ. – СПб.: Символ Плюс, 2008. –512 с., ил., ISBN 10: 5 93286 109 6, ISBN 13: 978 5 93286 109 7
2. Немет, Эви, Снайдер, Гарт, Хейн, Трент, Уэйли, Бэн. Unix и Linux: руководство системного администратора, 4-е изд. : Пер. с англ. — М.: ООО “И Д Вильямс”, 2012. — 1312 с. : ил. — Парал. тит. англ., ISBN 978-5-8459-1740-9 (рус.)

### Интернет-документы:

1. RFC 5321 SMTP, RFC 3501 IMAP - VERSION 4rev1, RFC 7208 SPF, RFC 6376 DKIM, RFC 7489 DMARC с сайта <https://www.rfc-editor.org/>
2. Rspamd. Spam filtering system. Доклад с конференции Highload++  
<https://rspamd.com/rspamd-highload.pdf>
3. Официальная группа поддержки Rspamd.  
<https://groups.google.com/forum/#!forum/rspamd>
4. Официальная документация с сайта Postfix:  
<http://www.postfix.org/documentation.html>
5. Официальная документация с сайта Dovecot:  
[http://wiki.dovecot.org/#Dovecot\\_configuration](http://wiki.dovecot.org/#Dovecot_configuration)
6. Официальный портал Zimbra:  
[https://wiki.zimbra.com/wiki/Main\\_Page](https://wiki.zimbra.com/wiki/Main_Page)
7. Веб-ресурсы о двойной доставке:  
<http://pjrllost.blogspot.ru/2012/11/smtp-delivery-to-two-mail-servers-via.html>  
<http://www.tech-notes.net/configure-dual-delivery-postfix>

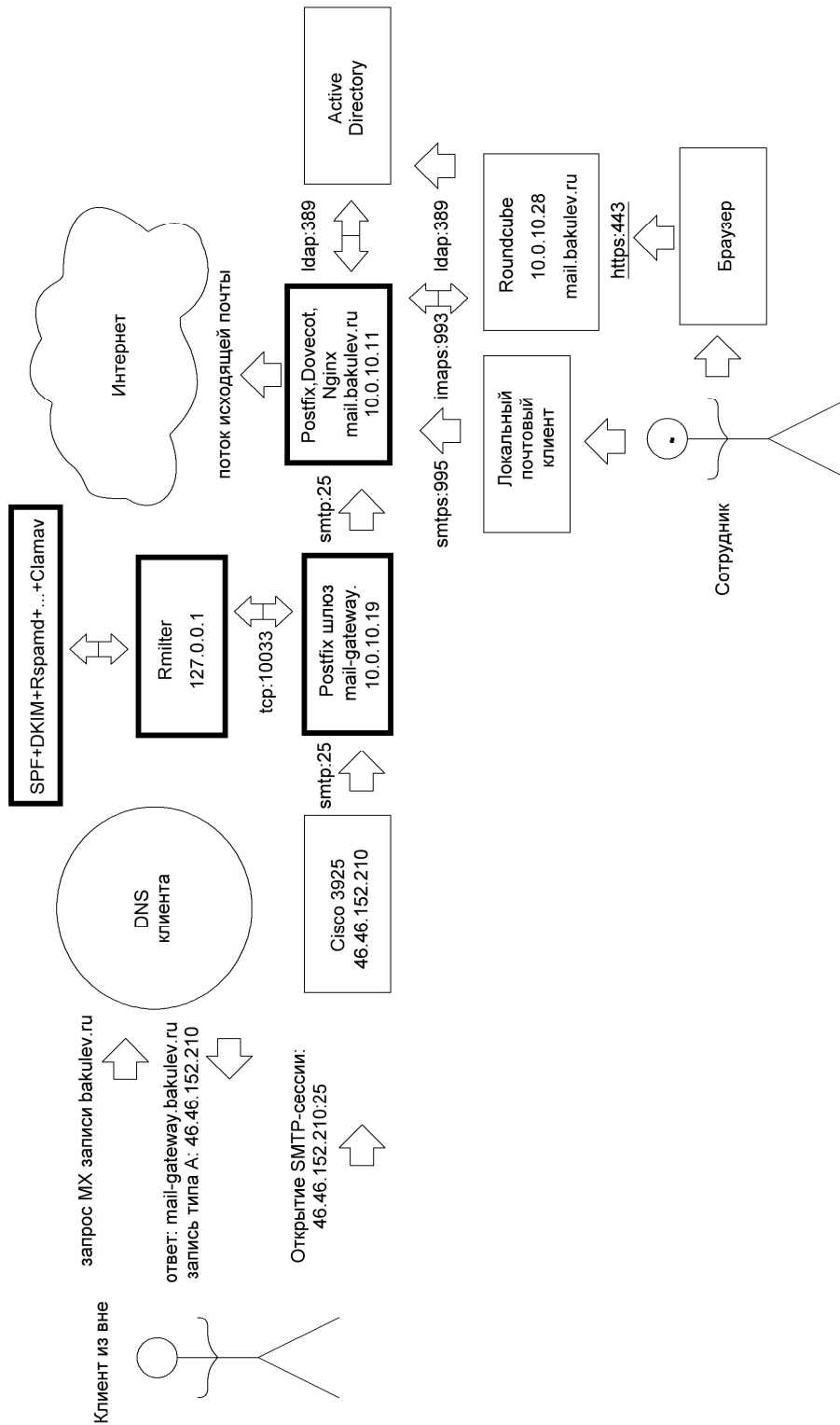
# Приложения

Приложение 1. Схема работы почты до модернизации.





## Приложение 2. Схема работы почты после модернизации.



### Приложение 3. Конфигурационный файл */etc/postfix/main.cf* МТА Postfix на новом

ПЛЮЗе:

```
myorigin = bakulev.ru
myhostname = mail-gateway
mydestination =
relay_domains = bakulev.ru
relayhost = [mail.bakulev.ru]:25
mailbox_size_limit = 0
message_size_limit = 31457280
recipient_delimiter = +
inet_interfaces = 10.0.10.19, 127.0.0.1
local_transport = error:local delivery is disabled

#Rmilter integration
smtpd_milters = inet:127.0.0.1:10033
milter_protocol = 6
milter_mail_macros = i {mail_addr} {client_addr} {client_name}
# skip mail without checks if milter will die
milter_default_action = accept

#SMTP restrictions
smtpd_delay_reject = yes
disable_vrfy_command = yes
strict_rfc821_envelopes = yes
smtpd_helo_required = yes

smtpd_error_sleep_time = 3
smtpd_soft_error_limit = 5
smtpd_hard_error_limit = 7

smtpd_client_restrictions =
    sleep 5
    reject_unknown_reverse_client_hostname

smtpd_helo_restrictions =
    reject_invalid_helo_hostname
    reject_non_fqdn_helo_hostname
    reject_unknown_helo_hostname
```

```
check_helo_access hash:/etc/postfix/my_helo
check_helo_access pcre:/etc/postfix/helo_checks.pcre
smtpd_sender_restrictions =
    reject_non_fqdn_sender
    reject_unknown_sender_domain
    check_sender_access hash:/etc/postfix/gtfo_s
    check_sender_access pcre:/etc/postfix/bad_domains.pcre
smtpd_recipient_restrictions =
#    check_client_access hash:/etc/postfix/client_whitelist
    reject_non_fqdn_recipient
    reject_unknown_recipient_domain
    reject_unauth_destination
    reject_unverified_sender
    reject_unverified_recipient
    check_recipient_access hash:/etc/postfix/gtfo_r
smtpd_data_restrictions =
    reject_unauth_pipelining
    reject_multi_recipient_bounce
```

Приложение 4. Часть конфигурационного файла /etc/postfix/master.cf МТА Postfix на новом почтовом сервере:

```
#доставка на mail.bakulev.ru
dualdelivery unix - n n - 5 pipe
    user=vmail argv=/opt/smtpdd/smtpdd.sh /var/spool/smtpdd ${sender} ${recipient} localhost:10026:q
mail.bakulev.ru:25:q
#доставка на себя
localhost:10026 inet n - n - - smtpd
    -o content_filter=
    -o receive_override_options=no_unknown_recipient_checks,no_header_body_checks,no_milters
#говорим демону smtpd использовать контент-фильтр
smtp inet n - n - - smtpd
    -o content_filter=dualdelivery
```

## Приложение 5. Исходный текст скрипта для двойной доставки:

```
#!/bin/bash
MSMTP="/usr/bin/msmtp"
TMPDIR="/var/tmp/smtpdd"
TMPFILENAME="mailqfile.$RANDOM.`date +%s`.$$"
LOCKFILE="$1/.smtpdd.lck"

EX_TEMPFAIL=75

printUsage()
{
echo "Usage: `basename $0` queue-directory sender recipient hostname[ip:port:mode [hostname[ip:port:mode ...]]"
echo "    : `basename $0` queue-directory qrun";
echo "Where mode is q for queuing delivery, d for just drop it or o to just queue the mail - defaults to q"
}

attemptDelivery()
{
    qd=$1
    mail=$2
    host=`basename $1`

    from=`cat $mail|cut -f 1 -d " "`
    to=`cat $mail|cut -f 2 -d " "`

    hostname=`echo $host|cut -f 1 -d :`
    port=`echo $host|cut -f 2 -d :`

    if [ ! -f $mail.body ]
    then
        echo "Cannot delivery mail, mail body is missing"
        exit $EX_TEMPFAIL
    fi

    #echo "atmpting delivery of $2 from $1 as $from and $to to host $hostname with port $port"
    $MSMTP --host $hostname --port $port -f $from $to < $mail.body
    if [ $? == 0 ]
    then
```

```

        echo "Message delivered, deleting"
        rm -f $mail $mail.body
    else
        echo "Message delivery failed, leaving in place"
    fi
}

queueRun()
{
for dirs in $1/*
do
    if [ -d $dirs ]
    then
        old_pwd=$PWD
        cd $dirs
        for mail in *.qf
        do
            if [ -f $mail ]
            then
                attemptDelivery "$dirs" "$mail"
            fi
        done
        cd $old_pwd
    fi
done
}

mainRun()
{
if [ ! -d $TMPDIR ]
then
    mkdir $TMPDIR
    if [ $? != 0 ]
    then
        echo "Tempdirectory configuration problem with $TMPDIR - cannot create directory"
        logger -p mail.error -t smtpdd "Temp directory configuration problem with $TMPDIR - cannot
create directory"

        exit $EX_TEMPFAIL
    fi
fi
}

```

```

        fi
    fi

chmod 0700 $TMPDIR

queuedir=$1
from=$2
to=$3

# should loop until it finds a unique filename
while [ -f $TMPDIR/$TMPFILENAME ]
do
    TMPFILENAME="$TMPFILENAME.$RANDOM"
    # echo "File exists already, generating new one, $TMPFILENAME"
done

cat > $TMPDIR/$TMPFILENAME

for host in ${@:4}
do
    #echo "attempting $2 to $3 for $host"

    hostname=`echo $host|cut -f 1 -d :`
    port=`echo $host|cut -f 2 -d :`
    mode=`echo $host|cut -f 3 -d :`
    if [ "x$mode" == "x" ]
    then
        mode="q"
    fi

    queueit="no"

    logger -p mail.info -t smtpdd "Attempting delivery of mail from $2 to $3 for host $hostname on $port
with mode of $mode"

    if [ "$mode" != "o" ]
    then
        # attempt real delivery

```

```

$MSMTP --host $hostname --port $port -f $from $to < $TMPDIR/$TMPFILENAME >
/dev/null 2>&1

RET=$?
if [ $RET != 0 ]
then
    if [ $RET != "75" ]
    then
        logger -p mail.info -t smtpdd "Mail delivery for $3 has failed"
        echo "Mail delivery for $3 has failed (unknown user?)"
        exit $RET
    fi
    # we failed to deliver ....
    # in queue mode, we deliver to the queue
    if [ "$mode" == "q" ]
    then
        #echo "Delivery of mail to $hostname on $port from $from to $to has failed,
but will be queued ($RET)"
        logger -p mail.warning -t smtpdd "Delivery of mail to $hostname on $port
from $from to $to has failed, but will be queued ($RET)"
        queueit="yes"
    fi

    # delete mode
    if [ "$mode" == "d" ]
    then
        logger -p mail.info -t smtpdd "Delivery of mail to $hostname on $port from
$from to $to has failed and will not be re-tried (delete mode)"
        randomtext="asdf"
    fi
else
    logger -p mail.info -t smtpdd "Delivery of mail to $hostname on $port from $from to
$to has succeeded"
    randomtext="asdf"
fi
else
    echo "In queue only mode, will queue"
    logger -p mail.info -t smtpdd "In queue-only mode, will queue mail until a queue run is
performed"

```



```

        queueit="yes"
    fi

    if [ $queueit == "yes" ]
    then
        echo "queueing"
        mkdir -p $queuedir/$hostname:$sport
        i=0
        while [ -f $queuedir/$hostname:$sport/mailf.$$.$i.qf ]
        do
            i=$(( $i + 1 ))
        done
        cp $TMPDIR/$TMPFILENAME $queuedir/$hostname:$sport/mailf.$$.$i.qf.body
        echo $from $to > $queuedir/$hostname:$sport/mailf.$$.$i.qf
    fi

done

rm -f $TMPDIR/$TMPFILENAME
}

if [ "$1" == "x" ]
then
    printUsage
    exit $EX_TEMPFAIL
fi

if [ ! -d $1 ]
then
    echo "queue-directory specified, $1, must exist"
    exit $EX_TEMPFAIL
fi

if ! touch $1/.fml
then
    echo "queue-directory specified, $1, must exist and be writable"
    exit $EX_TEMPFAIL
else

```

```
rm -f $1/.fml
```

```
fi
```

```
chmod 0700 $1
```

```
exec 8>> $LOCKFILE
```

```
if [ "x$2" == "xqrun" ]
```

```
then
```

```
if flock -n -e 8
```

```
then
```

```
queueRun "$1"
```

```
exit 0
```

```
else
```

```
logger -p mail.warning -t smtpdd "Queue directory locked, must exit from qrun"
```

```
exit $EX_TEMPFAIL
```

```
fi
```

```
fi
```

```
if [ "x$4" == "x" ]
```

```
then
```

```
printUsage
```

```
exit $EX_TEMPFAIL
```

```
fi
```

```
if flock -n -s 8
```

```
then
```

```
mainRun $@
```

```
else
```

```
logger -p mail.warning -t smtpdd "Cannot obtain shared lock - will not deliver right now"
```

```
exit $EX_TEMPFAIL
```

```
fi
```