|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

|  |  |
| --- | --- |
| **ФАКУЛЬТЕТ** | “Информатика и системы управления” |
| **КАФЕДРА** | “Информационная безопасность” |

**ОТЧЕТ О ДОЛГОСРОЧНОМ ПРОЕКТЕ**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Студент:** | Балашов Денис Васильевич, Барышев Михаил Сергеевич | | | |
|  | (фамилия, имя, отчество) | | | |
| **Курс:** | 2 | | **Группа:** | ИУ10-36 |
| **Специальность:** | | 10.05.03 | | |
| **Тип работы:** | | Долгосрочный проект | | |
| **Тема работы :** | | Развёртка стенда ELK для просмотра и анализа логов | | |
| **Предприятие:** | | МГТУ им. Н.Э. Баумана | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Студенты** |  |  | Д.В.Балашов  М.С.Барышев |

(подпись)

|  |  |  |  |
| --- | --- | --- | --- |
| **Преподаватель** |  |  | C.В.Лысюк |

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

Оглавление

**Введение1**

**Установка ELK4**

**Установка Kibana6**

**Установка Logstash8**

**Установка Filebeat12**

**Настройка безопасности ELK14**

**Создание кластера21**

**Заключение25**

**Цель работы**: Развернуть стенд состоящий из 2 нод, в котором подключены 2 машины, одна из них отправляет логи на сервер,а вторая хранит информацию о сборе и отправке.

**Задачи**:

1. Изучить процесс развёртки и настройки Elastic Kibana Logstash.
2. Запустить процесс отправки и сбора логов.
3. Настроить взаимодействие между собой 2 нод,находящихся в кластере.
4. Выяснить какое практическое применение может быть у нашего проекта и какие реальные задачи решаются с помощью ELK.

**Характеристики 1 машины:**

1. Процессор: Intel Core i7-4770 CPU 3.40GHz 4 ядра.
2. RAM 8 гб.

**Характеристики 2 машины:**

1. Процессор Intel core i5-1135G7 2.40 GHz 4 ядра.
2. RAM 4 гб.

**Ход работы:**

1. Подготовить систему для установки ELK:

а) определить кол-во необходимой RAM

б) проверить совместимость и версии

2.Установить и настроить:

а) Elastic

б) Kibana

в) Logstash

г) Filebeat

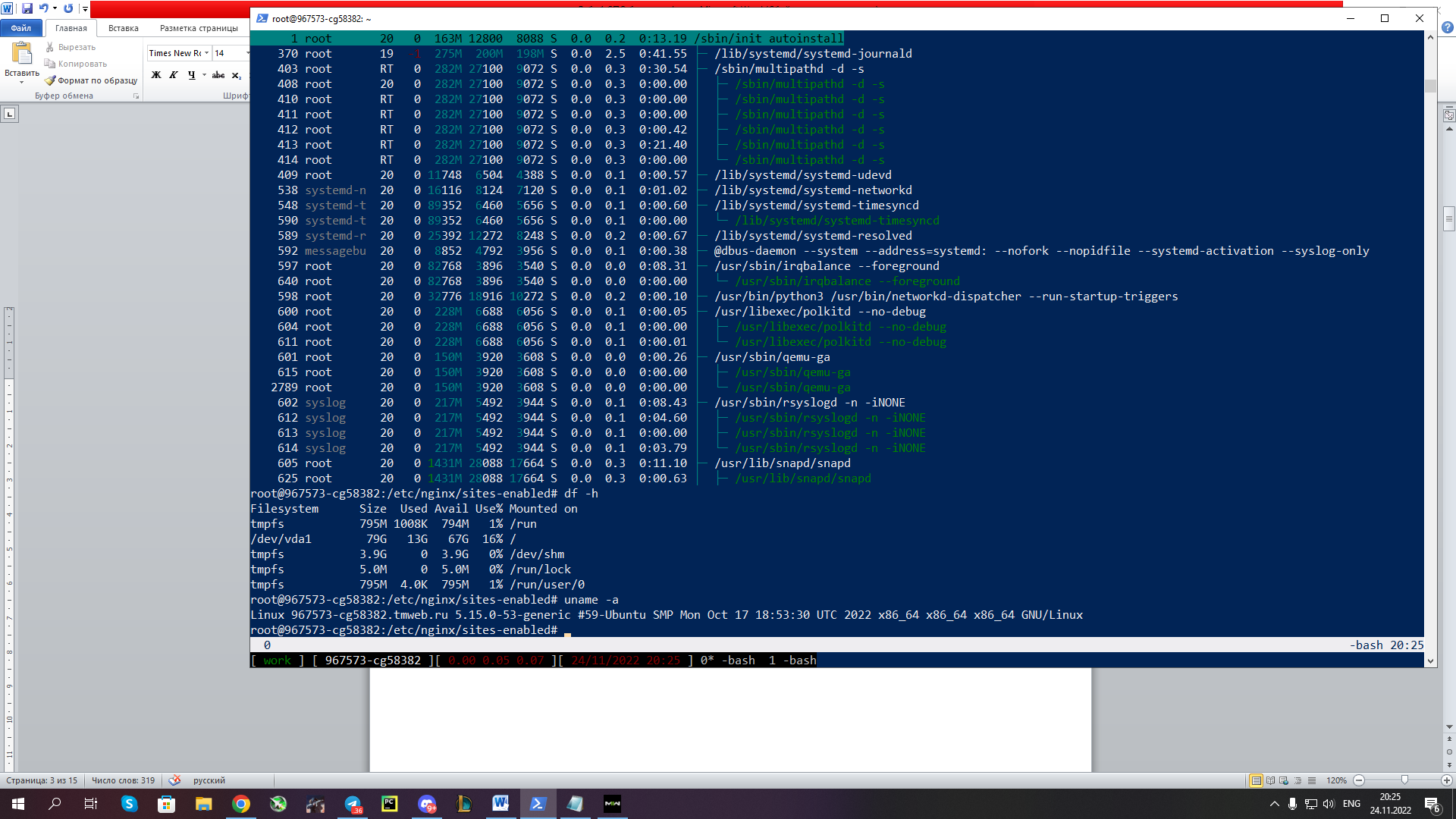
3.Создать и настроить 2 ноду (data node) для хранения всех данных.

**Подготовим систему для установки**

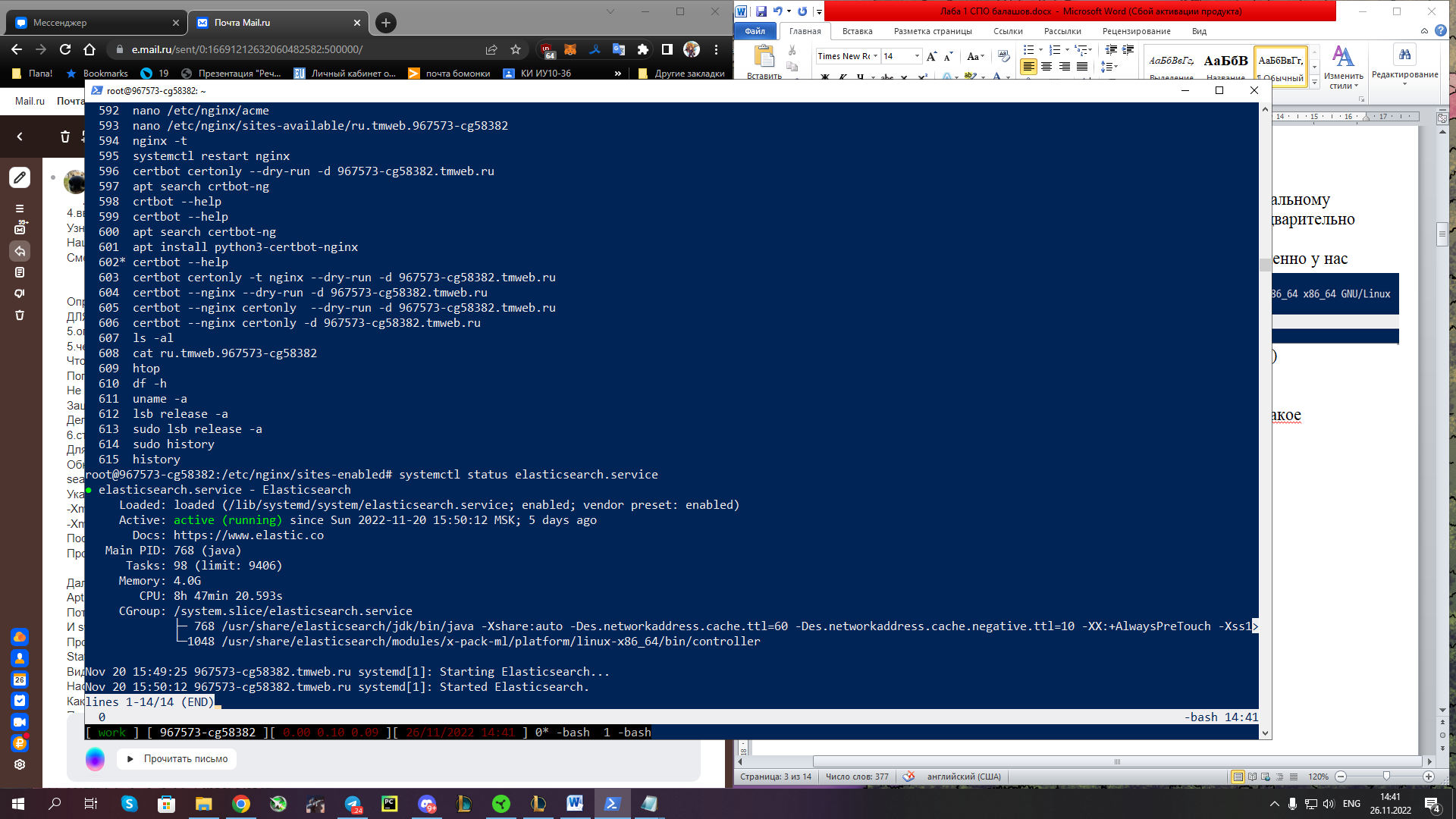
1) Для начала подключаемся к виртуальной машине через putty, либо же через powershell по ssh ключам, предварительно скопировав публичные ключи на сервер в папку authorized keys.

2) Далее вводим **uname –a** и узнаем, что мы в Linux.

3) Необходимо определить, что это apt система и выполнить apt update.



Очень важно знать версию, семейство и репозитории, подходящие при установке любой системы!



**Установка ELK**

Установим Elastic search через **apt install elastic search**,запустим и проверим, что он работает.

Видим, что у нас затрачивается слишком много оперативной памяти.

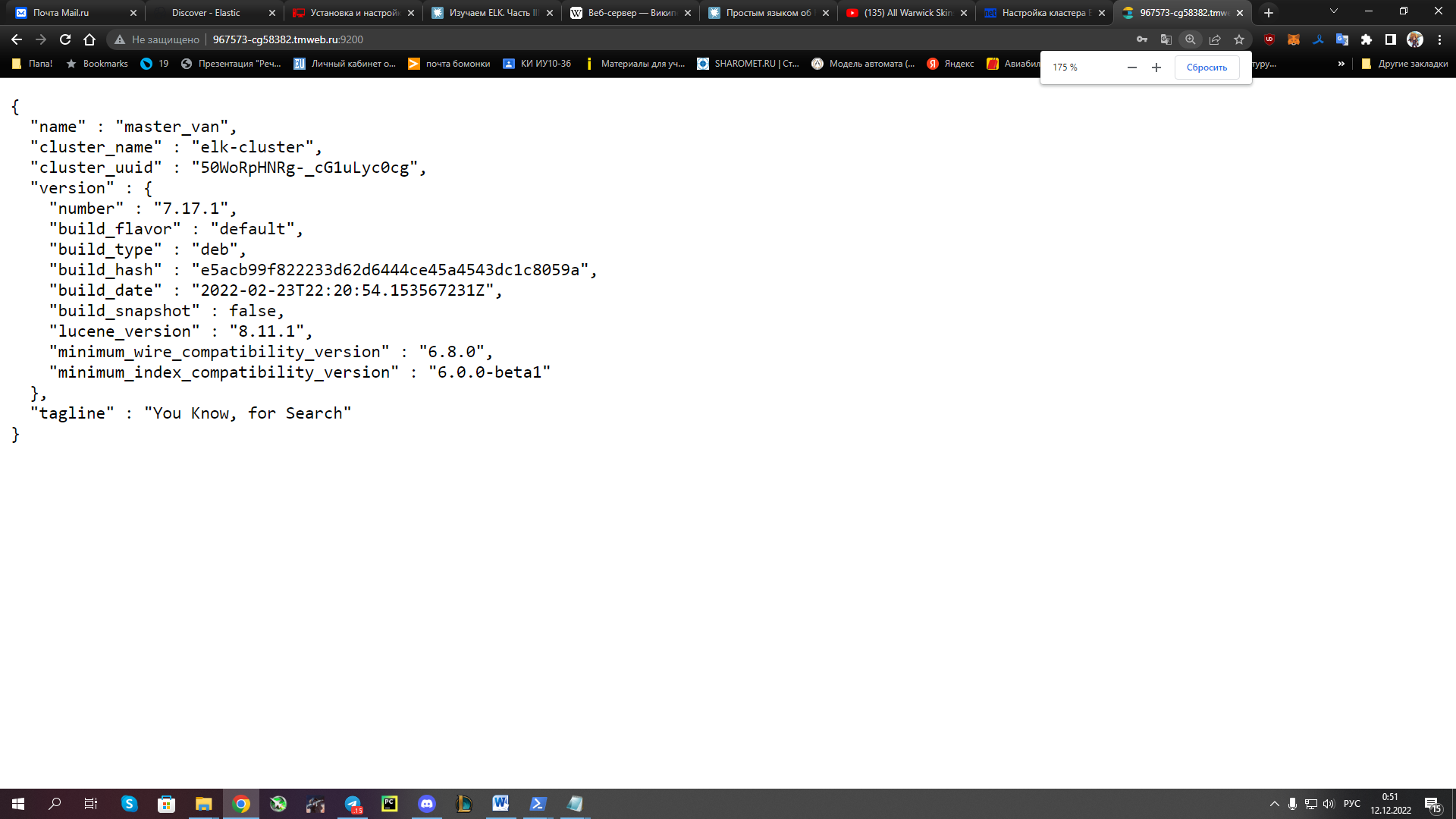
Для наших задач не будет критичным урезать количество RAM до 4 ГБ, зайдём в vim с помощью:

**vim /etc/elasticsearch/jvm.options.d/map.options** и укажем -Xms2g -Xmx2g.

После этого перезапускаем ElasticSearch и проверим что настройки применились.

Проверим, что ElasticSearch нормально работает выполним запрос о его статусе.

В браузере наберём 94.228.120.78:9200 и увидим, что всё работает.



“name” - имя ноды, которой принадлежит этот IP-адрес.

“cluster\_name” - имя кластера, куда входит нода.

“cluster\_uuid” – уникальный идентификатор набора реплик uuid.

**“**number” – версия Elasticsearch, установленная на данной машине.

“build\_flavor” – функциональность/наличие доп. возможностей Elasticsearch.

“build\_type” : deb – тип установленных пакетов, в нашем случае deb пакеты.

“build\_flavor” – хэш последнего коммита git в этом релизе.

“build\_date” – дата последних внесённых изменений в версию.

“build\_snapshot” : false – версия снэпшота, в нашем случае снэпшота нет.

“lucene\_version” – версия JAVA-библиотеки (lucene).

“minimum\_wire\_compatibility\_version” – способность связываться с узлами других версий.

“minimum\_index\_compatibility\_version” – параметр означающий

минимальную требуемую версию узлов для считывания данных.

“tagline” : “You know, for Search” – устоявшийся слоган компании

ElasticSearch.

**Установка KIBANA**

Установка Kibana – необходимо подключить репозиторий и используем deb пакет. Далее добавляем публичный ключ:

# wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | apt-key add -

Добавляем репозиторий Kibana:

# echo "deb https://artifacts.elastic.co/packages/8.x/apt stable main" | tee -a /etc/apt/sources.list.d/elastic-8.x.list

Запускаем установку Kibana:

# apt update && apt install kibana

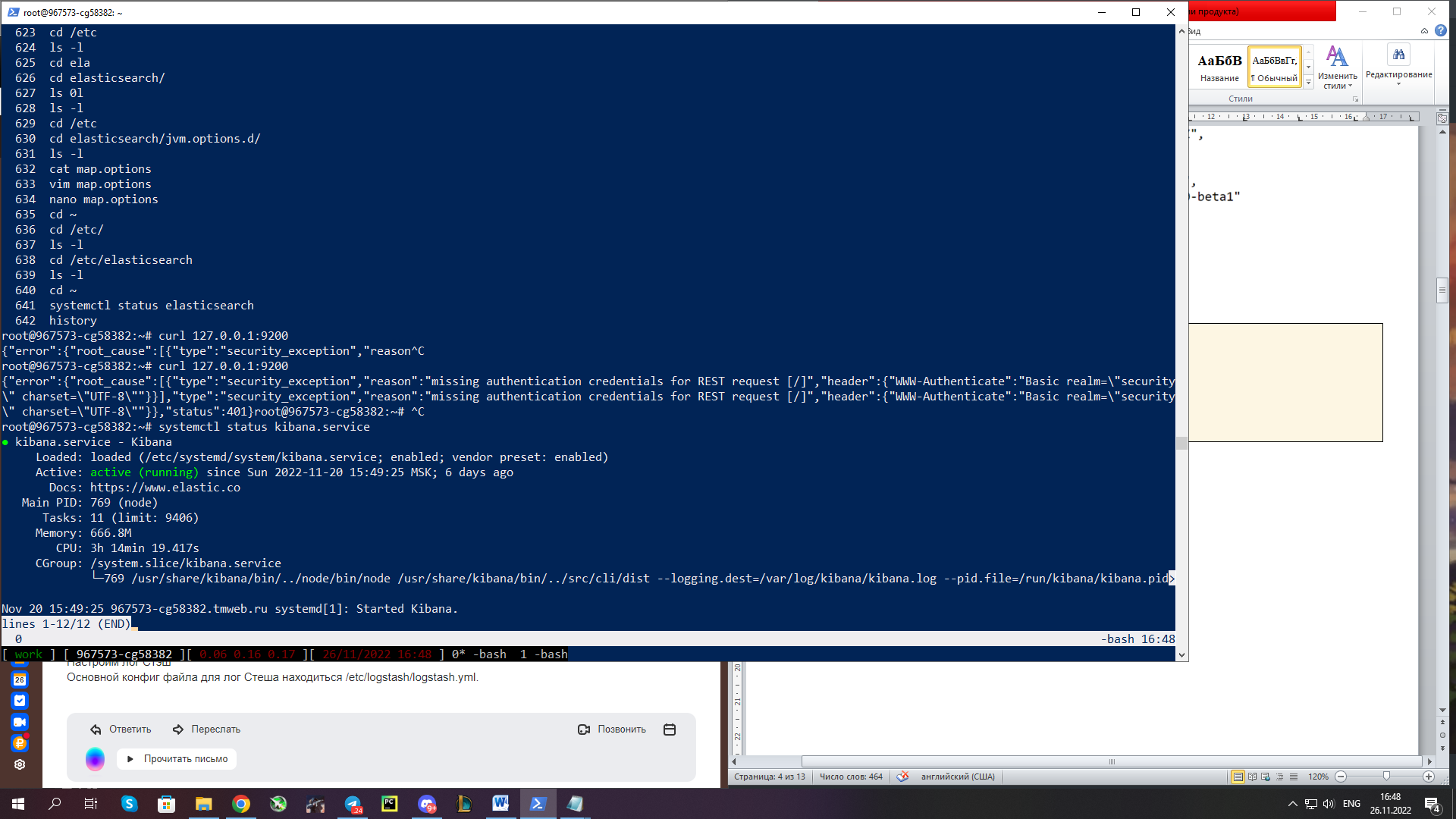
Добавляем Kibana в автозагрузку и запускаем:

# systemctl daemon-reload

# systemctl enable kibana.service

# systemctl start kibana.service

Проверяем состояние Kibana через status kibana service и видим, что все работает.

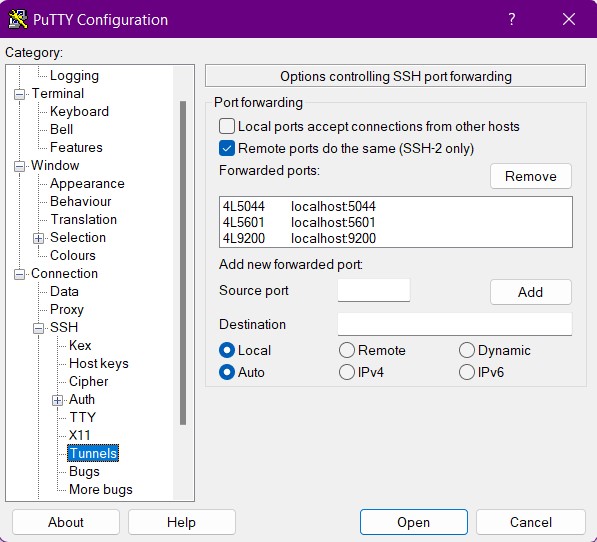


Далее необходимо настроить проброс портов для доступа к веб интерфейсу Kibana.

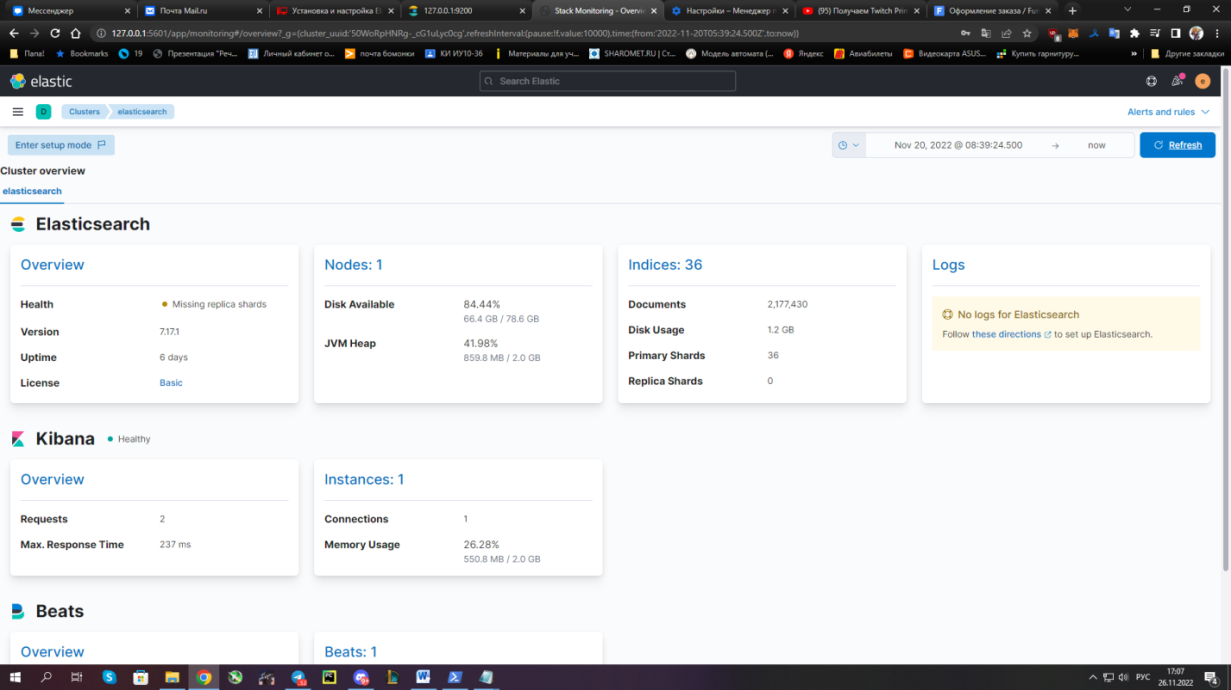
В случае если происходит подключение через PowerShell, то используется следующая команда:

ssh -L 9200:127.0.0.1:9200 -L 5601:127.0.0.1:5601 -L 5044:127.0.0.1:5044 [root@94.228.120.78](mailto:root@94.228.120.78)

Если подключение происходит через putty, тогда в настройках указываются следующие параметры:



Таким образом мы настроили проброс портов с удалённого на локальный(они совпадают) и получили доступ к Kibana:



**Установка и настройка Logstash**

# apt install logstash

Настройка Logstash осуществляется путём создания **input,output и filter.conf** файлов в директории **/etc/logstash/conf.d.**

Первая конфигурация в файлe input.conf будет выглядеть вот так:

**input {**

**beats {**

**port => 5044**

**}**

**}**

Необходимо указать, что информация принимается на 5044 порт.

Далее нужно указать, куда будут передаваться данные, для этого создаётся конфигурация **output.conf,** который будет описывать передачу данных в Elasticsearch:

**output {**

**elasticsearch {**

**user => "logstash\_nginx\_rw"**

**password => "glHcR5zqNnlDhXIAYtsCp1fCeuMYi0ii"**

**hosts => "localhost:9200"**

**index => "nginx-%{+YYYY.MM.dd}"**

**}**

**#stdout { codec => rubydebug }**

**}**

Итог: мы настроили передачу всех данных в Elasticsearch под указанным индексом с маской в виде даты. Разбиение индексов по дням и по типам данных удобно с точки зрения управления данными.

Далее настраивается конфигурация с описанием обработки логов. Создаём конфигурацию **filter.conf** :

**filter {**

**if [type] == "nginx\_access" {**

**grok {**

**match => { "message" => "%{IPORHOST:remote\_ip} - %{DATA:user} \[%{HTTPDATE:access\_time}\] \"%{WORD:http\_method} %{DATA:url} HTTP/%{NUMBER:http\_version}\" %{NUMBER:response\_code} %{NUMBER:body\_sent\_bytes} \"%{DATA:referrer}\" \"%{DATA:agent}\"" }**

**}**

**}**

**date {**

**match => [ "timestamp" , "dd/MMM/YYYY:HH:mm:ss Z" ]**

**}**

**geoip {**

**source => "remote\_ip"**

**target => "geoip"**

**add\_tag => [ "nginx-geoip" ]**

**}**

**}**

Первое, что делает этот фильтр - парсит логи nginx с помощью **grok**, если указан соответствующий тип логов, и выделяет из лога ключевые данные, которые записывает в определенные поля, чтобы потом с ними было удобно работать.

Далее используется модуль **date** для того, чтобы выделять дату из поступающих логов и использовать ее в качестве даты документа в Elasticsearch. Делается это для того, чтобы не возникало путаницы, если будут задержки с доставкой логов. В системе сообщения будут с одной датой, а внутри лога будет другая дата.

В конфигурации используется **geoip** фильтр, который на основе ip-адреса, полученного ранее с помощью фильтра **grok**, и записывается в поле **remote\_ip**, определяет географическое расположение. Он добавляет новые метки и записывает туда географические данные.

Для его работы используется база данных из файла **/usr/share/logstash/vendor/bundle/jruby/2.5.0/gems/logstash-filter-geoip-6.0.3-java/vendor/GeoLite2-City.mmdb**.Она будет установлена вместе с Logstash.

Закончили настройку Logstash. Запускаем его:

# systemctl start logstash.service

**Установка и настройка FileBeat**

# apt install filebeat

После установки создаётся конфигурация **/etc/filebeat/filebeat.yml** для отправки логов в Logstash:

**- type: log**

**enabled: true**

**paths:**

**- /var/log/nginx/\*access.log**

**fields:**

**type: nginx\_access**

**fields\_under\_root: true**

**scan\_frequency: 5s**

**- type: log**

**enabled: true**

**paths:**

**- /var/log/nginx/\*error.log**

**fields:**

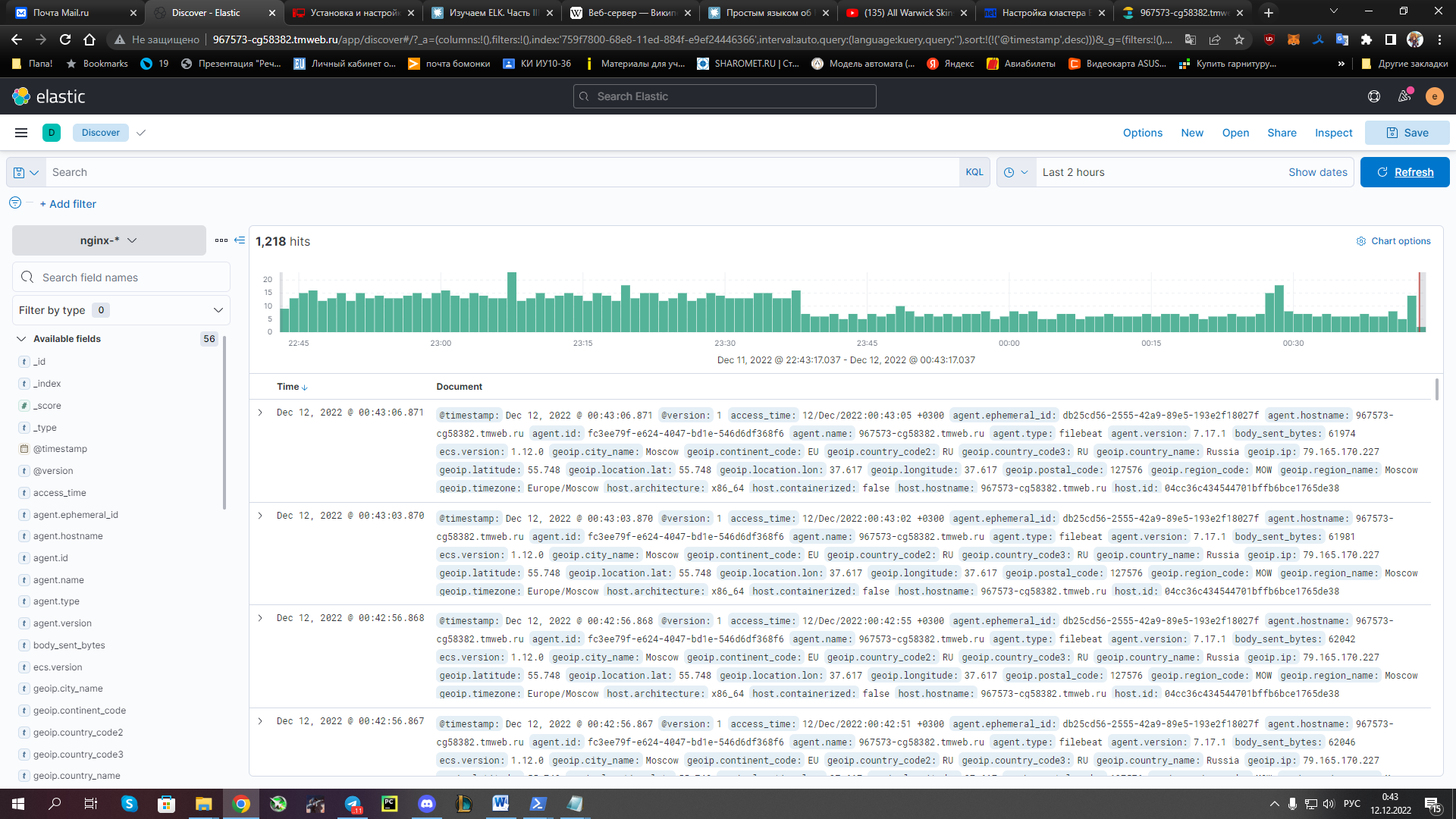
**type: nginx\_error**

**fields\_under\_root: true**

**scan\_frequency: 5s**

В данной конфигурации разделяются логи **access** и **error** с помощью отдельного поля type, куда записывается соответствующий тип лога **nginx\_access** или **nginx**\_**error**. В зависимости от типа меняются правила обработки в Logstash. Плюс, мы включили мониторинг и для этого указали адрес Elasticsearch, куда Filebeat передает данные мониторинга напрямую.

Теперь активируем этот мониторинг в веб-интерфейсе **Kibana.**



Здесь настраивается раздел Discover для логов, которые пишет **nginx**-сервер.

Настройка сбора и получения логов с **nginx**-сервера завершена.

**Настройка безопасности ELK**

Зачем вообще настраивать безопастность Elasticsearch?

Если установить Elasticsearch и никак его не защищать, то доступ ко всем индексам будет открыт для всех желающих. Или тех, кто может пользоваться curl. Чтобы этого избежать, в Elasticsearch есть ролевая модель, которая доступна начиная с подписки уровня Basic (она бесплатна). Схематически выглядит так:



* Пользователи — это все, кто может авторизоваться с использованием учётных данных.
* Роль — это набор прав.
* Права — это набор привилегий.
* Привилегии — это разрешения на запись, чтение, удаление и т.д.
* Ресурсы — это индексы, документы, поля, пользователи и другие субъекты хранилища.

Чтобы активировать безопасность в настройках Elasticsearch, нужно добавить в конфигурационный файл **elasticsearch/elasticsearch.yml** новую строку:

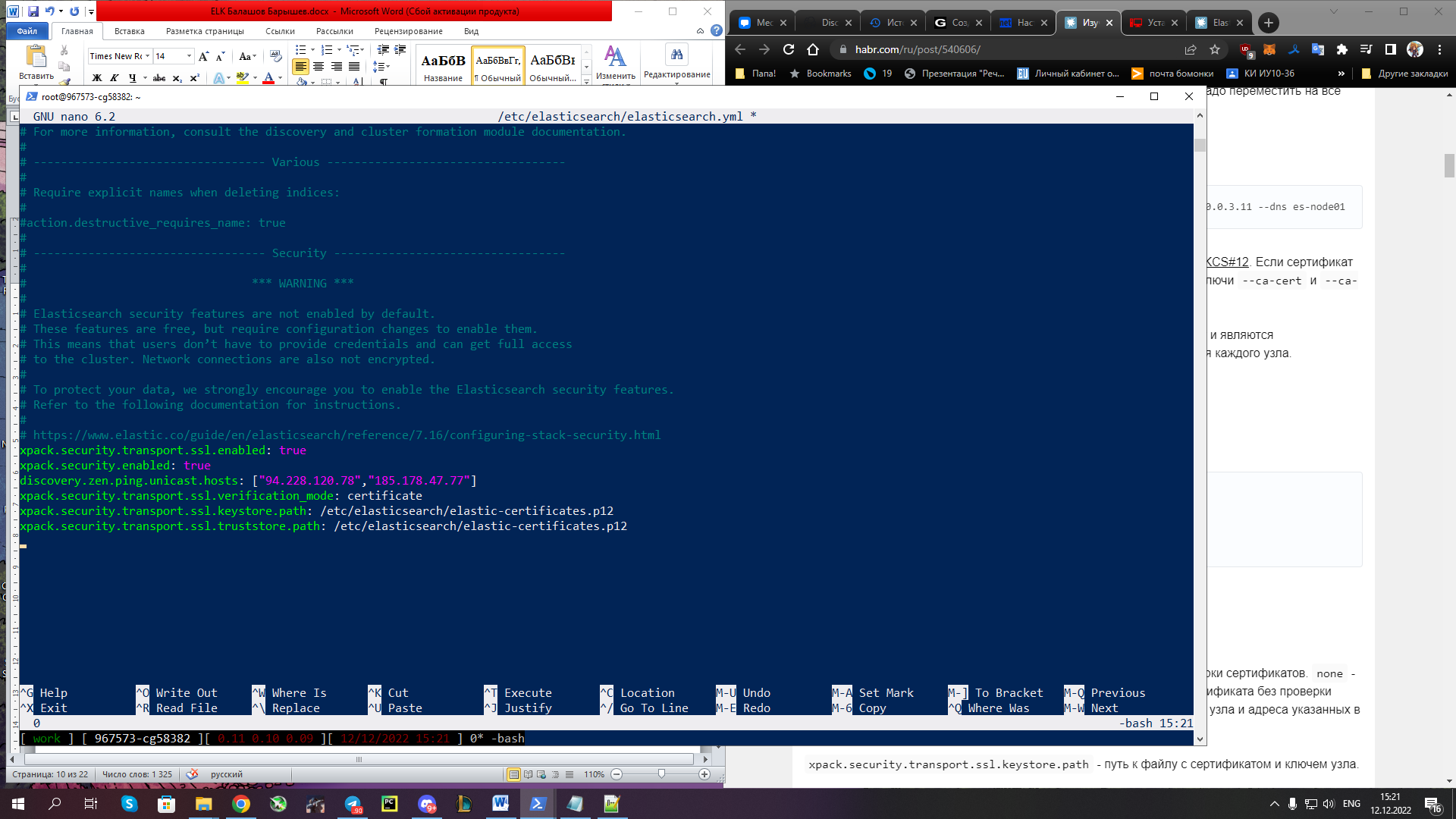
**xpack.security.enabled: true**

Следующим шагом необходимо настроить шифрование трафика между узлами Elasticsearch. Для этого создадим СА сертификаты.  
 С помощью данной команды создадим архив с **.crt** и **.key** файлами:

**/usr/share/elasticsearch/bin/elasticsearch-certutil cert --ca-cert /root/certificates**

**/root/certificates/control/ca/ca.crt –ca-key /root/certificates/control/ca/ca.key**

Ключ **--ca** указывает путь к корневому сертификату CA в формате PKCS#12. Если сертификат и ключ были получены в PEM формате, то необходимо использовать ключи **--ca-cert** и **--ca-key** соответственно.



Далее необходимо перейти в конфигурацию Elasticsearch и добавить туда верификацию по SSL сертификатам.

**xpack.security.transport.ssl.verification\_mode: certificate** - выполняется проверка сертификата без проверки имени узла и IP адреса.

**xpack.security.transport.ssl.keystore.path**  - путь к файлу с сертификатом и ключем узла.

**xpack.security.transport.ssl.truststore.path** - путь к доверенному сертификату (CA).

Теперь перезапускаем Elastic и смотрим чтобы у нас ничего не сломалось и не упало.

Сейчас нам нужно настроить аутентификацию по паролям для доступа пользователей нашего ELK стэка.

Elasticsearch имеет несколько [встроенных пользователей](https://www.elastic.co/guide/en/elasticsearch/reference/current/built-in-users.html#built-in-users):

|  |  |
| --- | --- |
| **Пользователь** | **Описание** |
| elastic | Суперпользователь |
| kibana\_system | Используется для коммуникации между Kibana и Elasticsearch |
| logstash\_system | Пользователь, которого использует Logstash сервер, когда сохраняет информацию в Elasticsearch |
| beats\_system | Пользователь, которого использует агент Beats, когда сохраняет информацию в Elasticsearch |
| apm\_system | Пользователь, которого использует APM сервер, когда сохраняет информацию в Elasticsearch |

Далее генерируются пароли для пользователей с помощью следующей команды:

**./bin/elasticsearch-setup-passwords auto**

**Initiating the setup of passwords for reserved users elastic,apm\_system,kibana,kibana\_system,logstash\_system,beats\_system,remote\_monitoring\_user.**

**The passwords will be randomly generated and printed to the console.**

**Please confirm that you would like to continue [y/N]y**

**Changed password for user apm\_system**

**PASSWORD apm\_system = NtvuRYhwbKpIEVUmHsZB**

**Changed password for user kibana\_system**

**PASSWORD kibana\_system = ycXvzXglaLnrFMdAFsvy**

**Changed password for user kibana**

**PASSWORD kibana = ycXvzXglaLnrFMdAFsvy**

**Changed password for user logstash\_system**

**PASSWORD logstash\_system = vU3CuRbjBpax1RrsCCLF**

**Changed password for user beats\_system**

**PASSWORD beats\_system = c9GQ85qhNL59H2AXUvcA**

**Changed password for user remote\_monitoring\_user**

**PASSWORD remote\_monitoring\_user = wB320seihljmGsjc29W5**

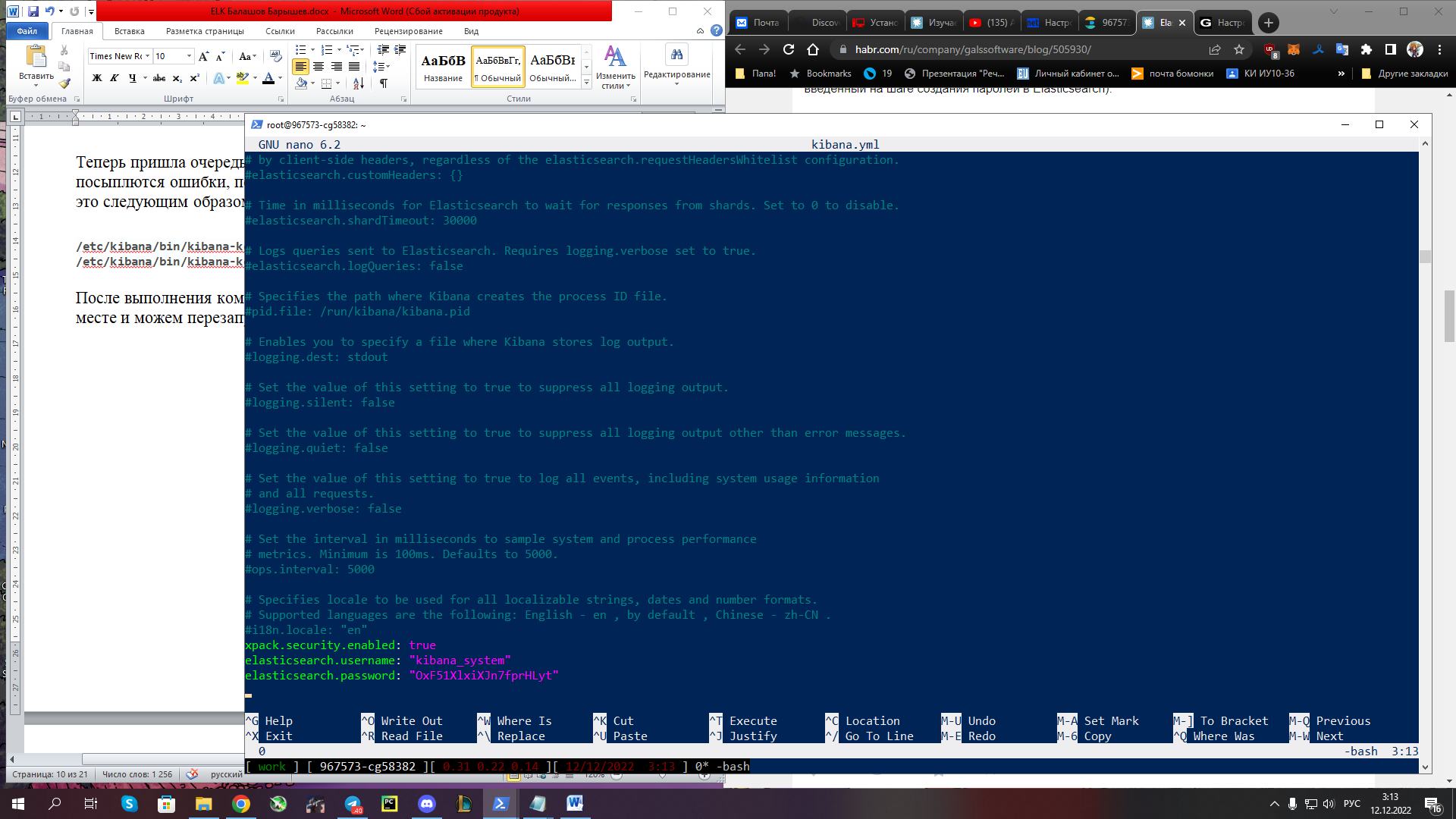
**Changed password for user elastic**

**PASSWORD elastic = iOrMTBbfHOAkm5CPeOj7**

Сохраняем пароли в файле и обнаруживаем, что доступ в веб интерфейс Kibana потерян. Теперь важно создать хранилище ключей. Делается, это следующим образом:

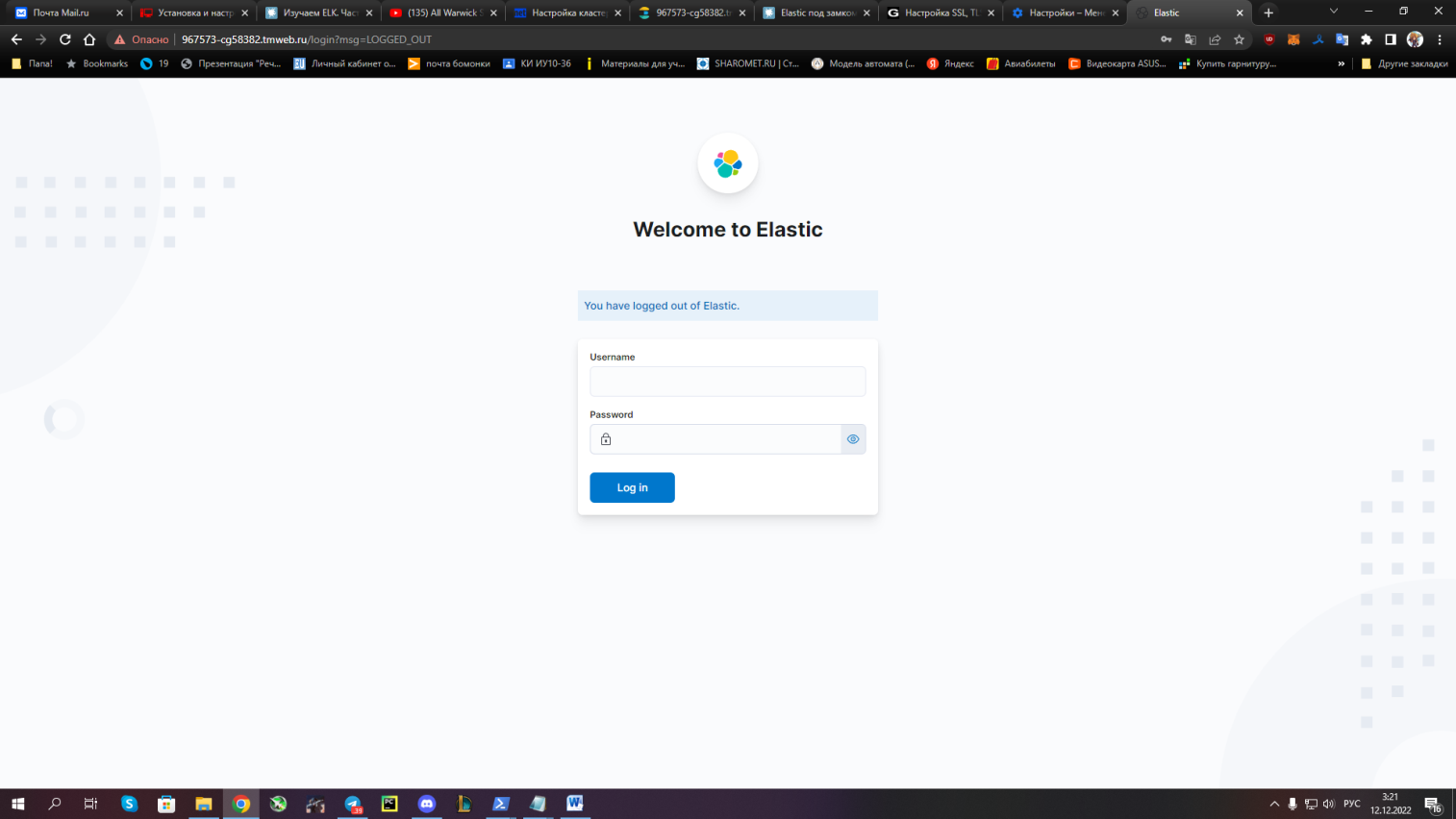
**/etc/kibana/kibana.yml add elasticsearch.username**

**/etc/kibana/kibana.yml add elasticsearch.password**



После выполнения команд необходимо зайти в конфигурацию Kibana, убедиться, что всё на месте, после чего следует перезапустить Kibana.

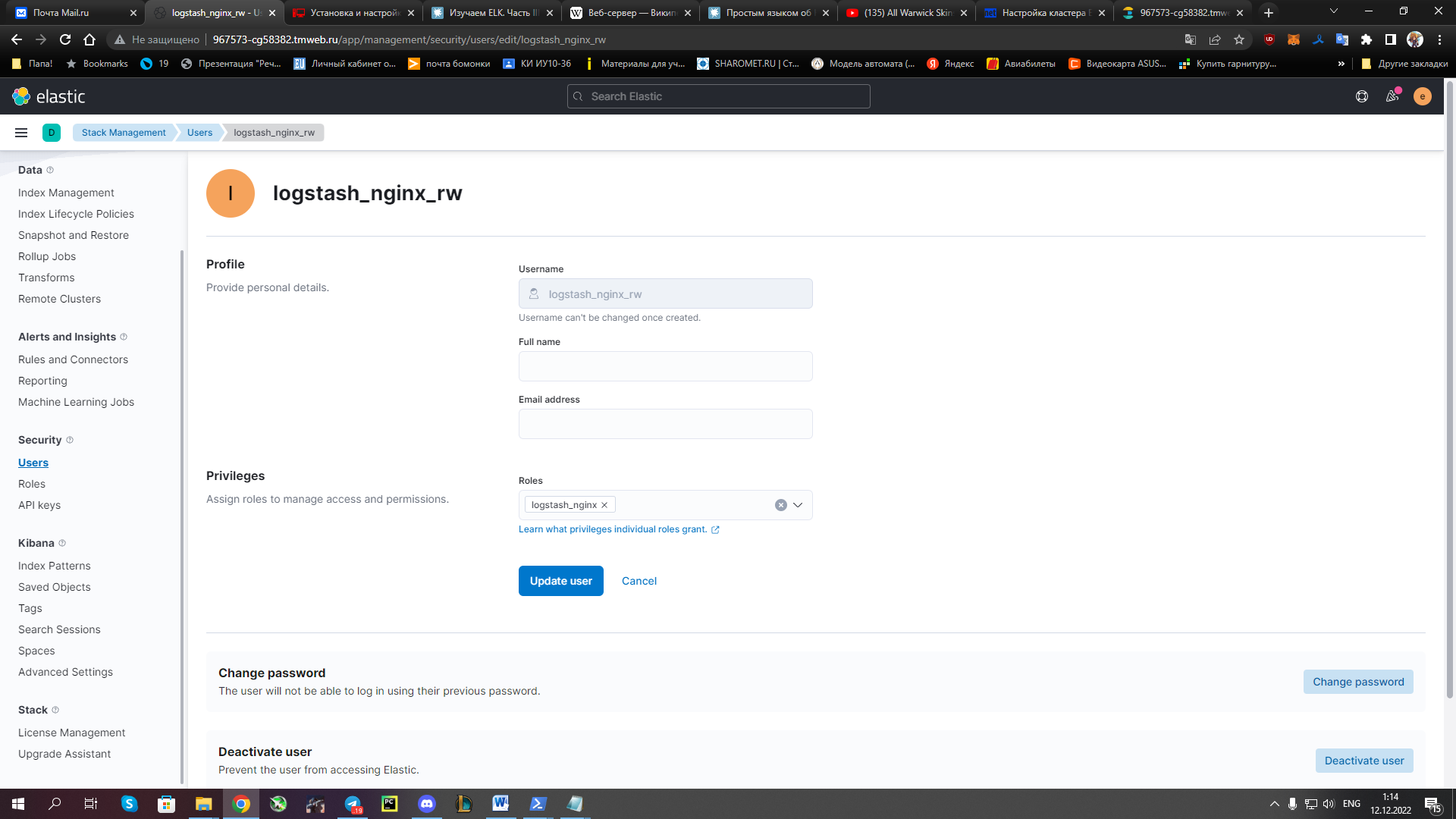
Теперь открываем Kibana в браузере и видим, что она запрашивает логин и пароль:





Следующим этапом создаётся и настраивается отдельный пользователь с правами на запись в нужные индексы. Перейдём в Kibana -> stack management и добавим там роль “logstash\_nginx”.

Далее переходим в пункт Users, добавляем нашего пользователя и назначаем ему созданную ранее роль.



Необходимо перейти в конфиг Logstash, а именно в output.conf и добавить учётные данные для доступа в кластер.

**output {**

**elasticsearch {**

**user => "logstash\_nginx\_rw"**

**password => "glHcR5zqNnlDhXIAYtsCp1fCeuMYi0ii"**

**hosts => "http://94.228.120.78:9200"**

**index =>"nginx-%{+YYYY.MM.dd}"  
 cacert => "/etc/logstash/certs/http\_ca.crt"**

**}**

**}**

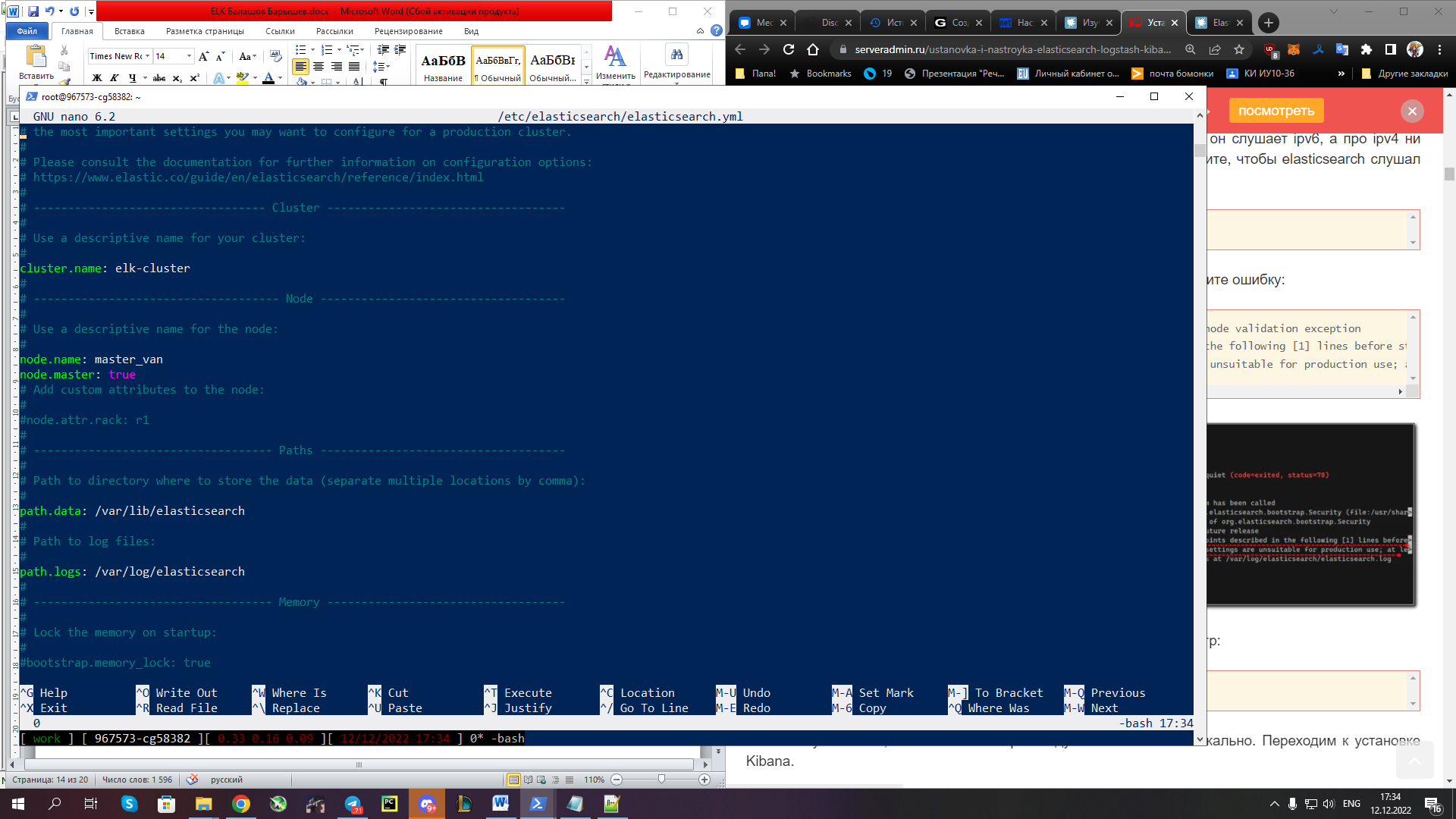
В строке hosts указываем IP нашего сервера и после этого данные, как и прежде, начнут поступать из Logstash в Elasticsearch, только теперь уже с авторизацией по пользователю и паролю.

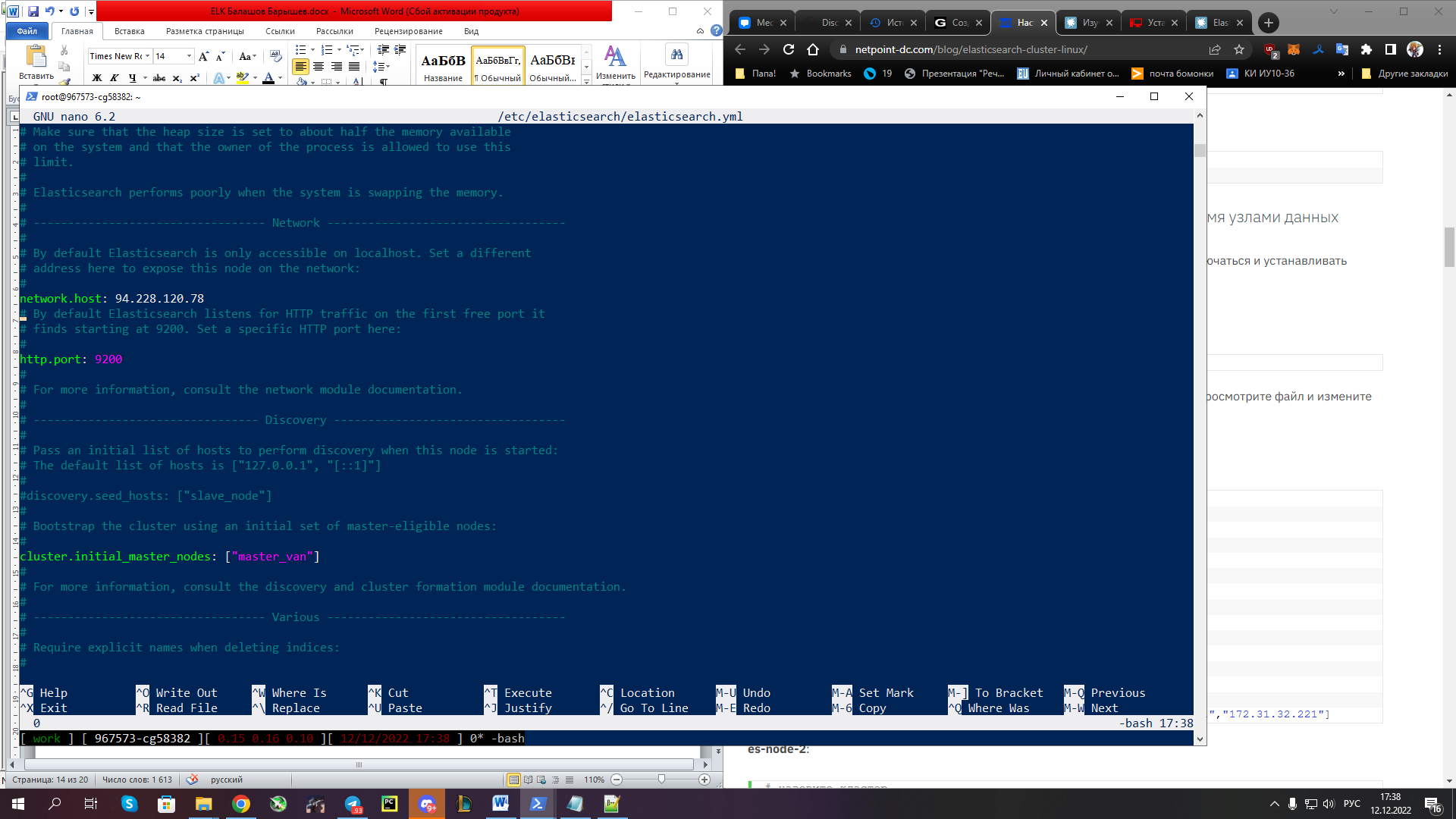
**Добавление 2 ноды в кластер**

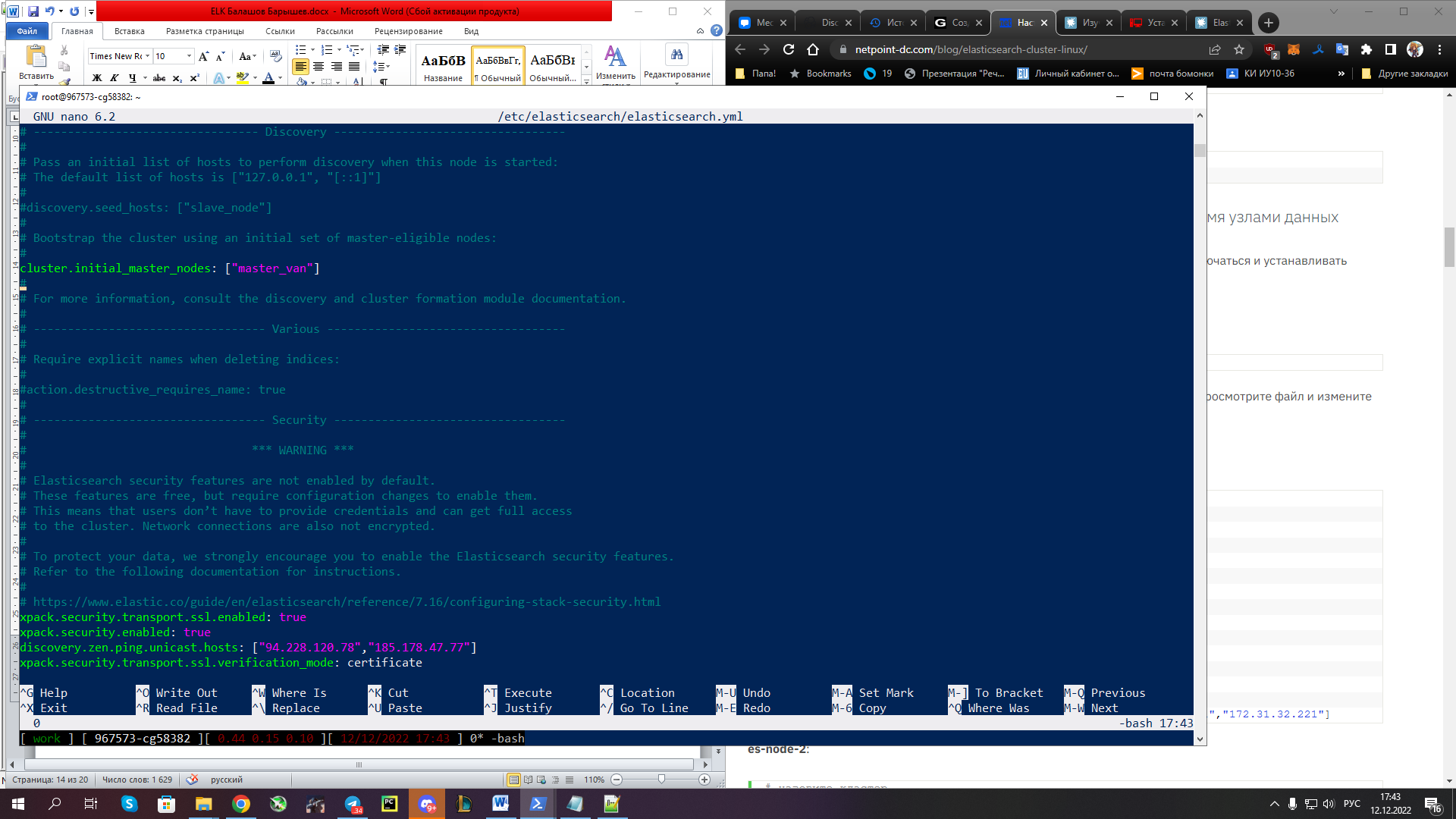
Одной из задач нашей проектной работы являлось создание кластера из двух нод. Для начала рассмотрим конфигурацию master-node: имя нашего кластера, затем имя ноды и обозначение, что именно эта нода будет master.

Далее в строке network.host указываем IP сервера, а в строке http.port указывается default порт Elasticsearch: 9200.

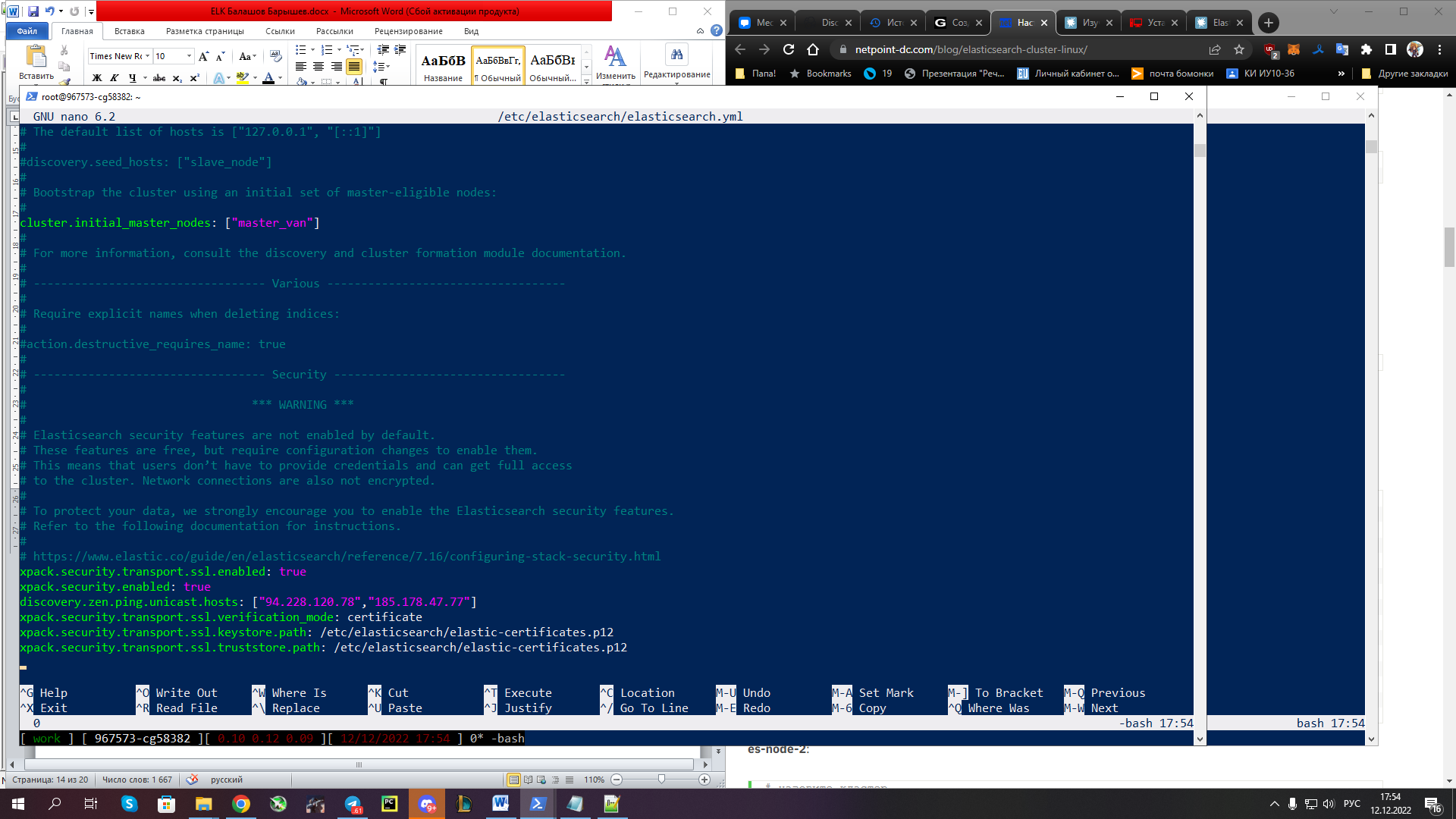
Если не указать параметр, подтверждающий мастер-ноду, то при запуске кластера может быть так, что обе ноды определят себя как master.



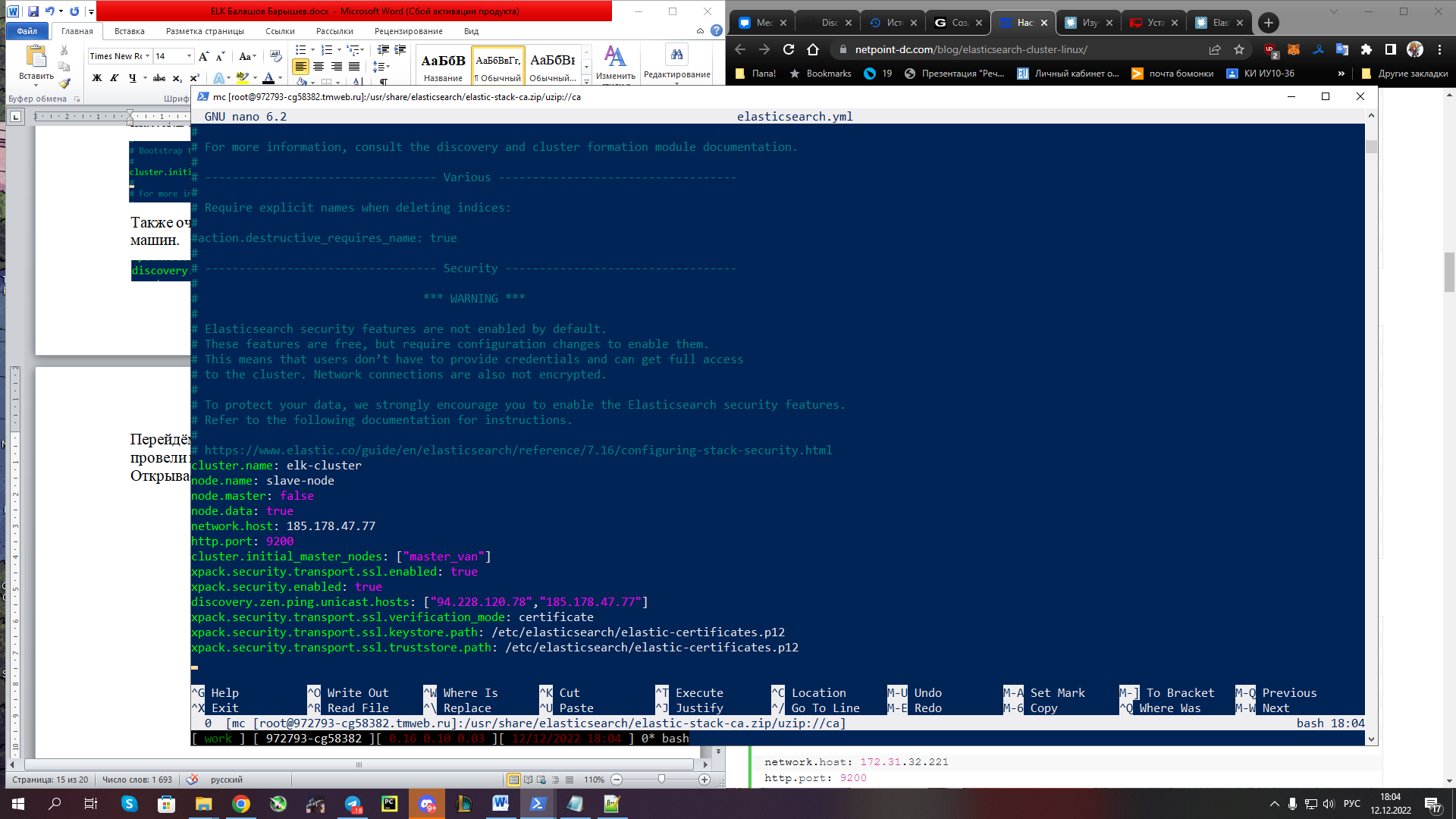




Также очень важной настройкой является добавление IP-адресов 2 машин.



Перейдём на 2 машину, на которой предварительно уже скачали Elasticsearch и провели первичную настройку.



Открываем конфигурацию Elasticssearch и добавляем следующие параметры:

**node.master: false**

**node.data: true**

Так как мы несколькими пунктами ранее создавали сертификаты аутентификации на 1 сервере, то их необходимо добавить и на этот сервер.

Копируем сертификаты с помощью данной команды:

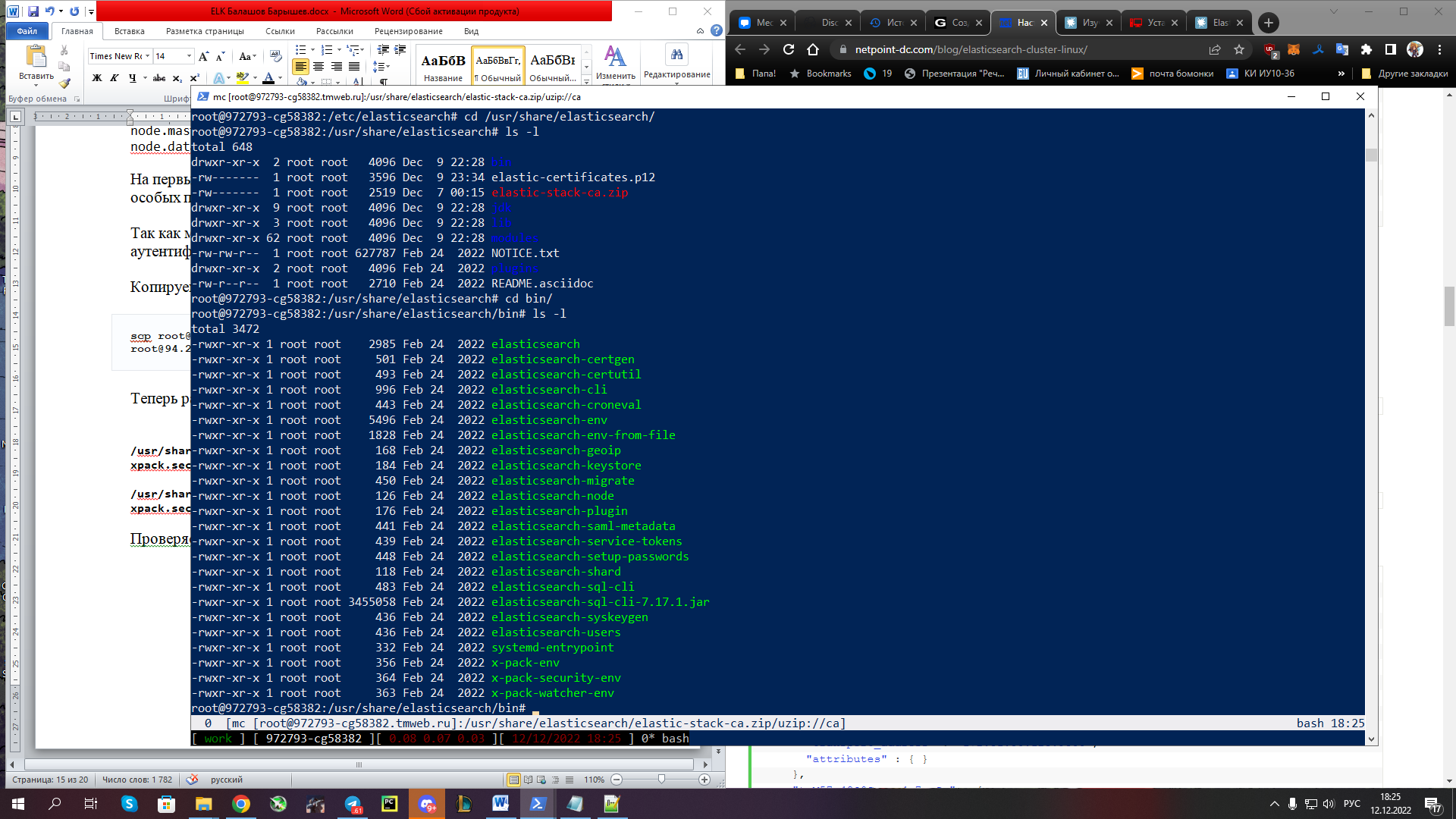
**Scp root@185.178.47.77:/usr/share/elasticsearch/elastic-stack-ca.zip root@94.228.120.78:/usr/share/elasticsearch/**

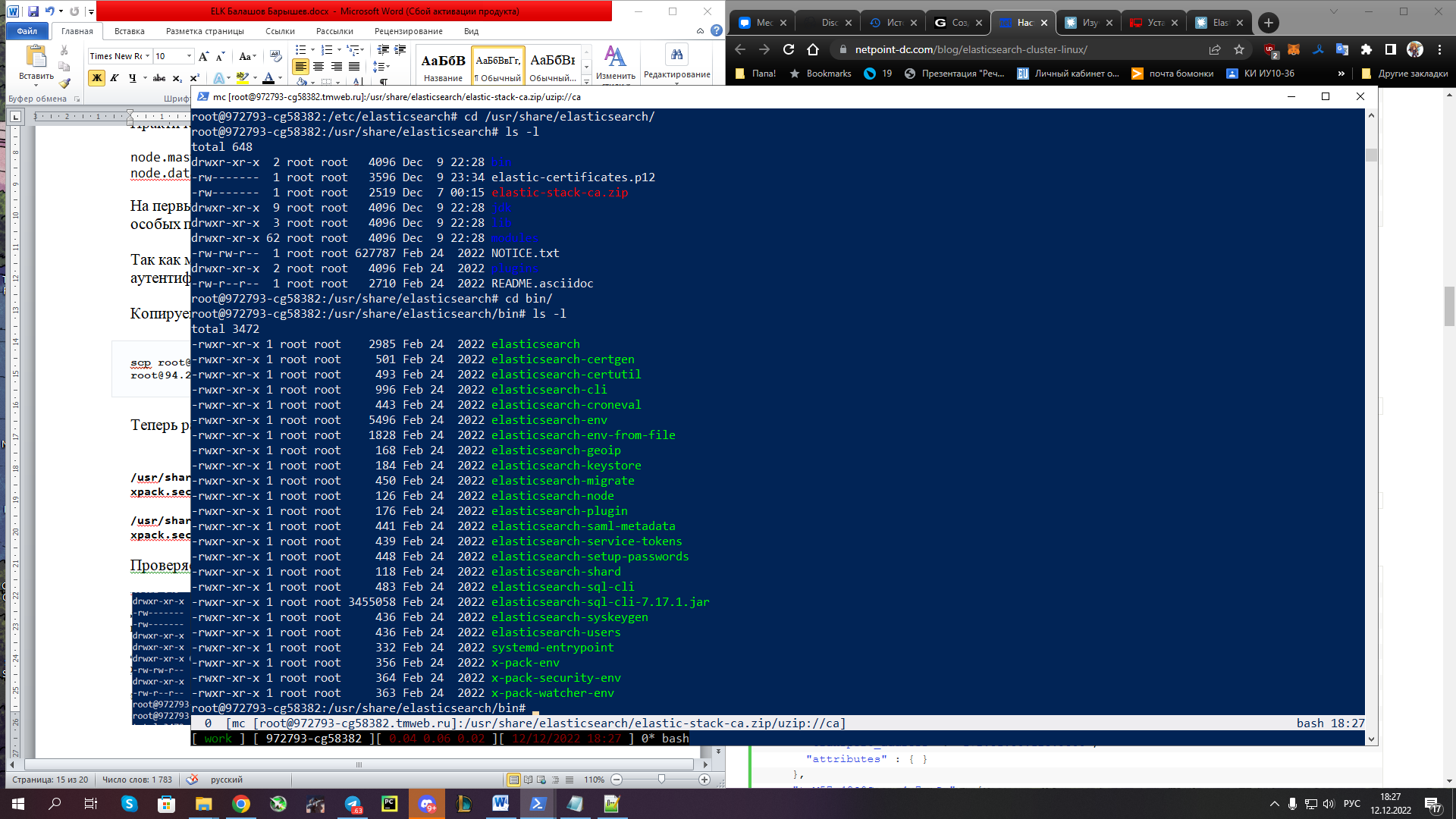
Теперь распаковываем архив и добавляем их в директорию с сертификатами.

**/usr/share/elasticsearch/bin/elasticsearch-keystore add xpack.security.transport.ssl.keystore.secure\_password**

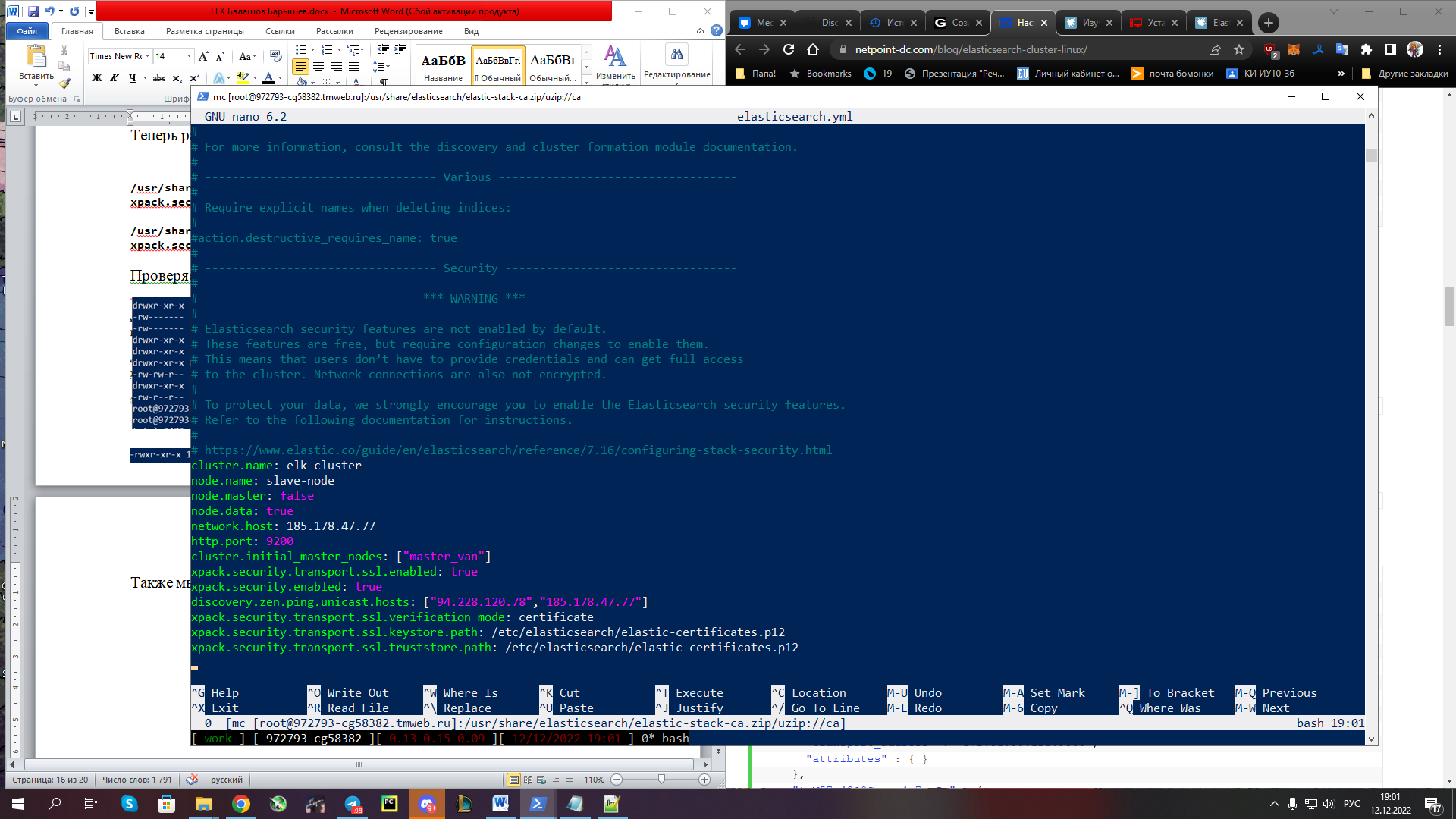
**/usr/share/elasticsearch/bin/elasticsearch-keystore add xpack.security.transport.ssl.truststore.secure\_password**

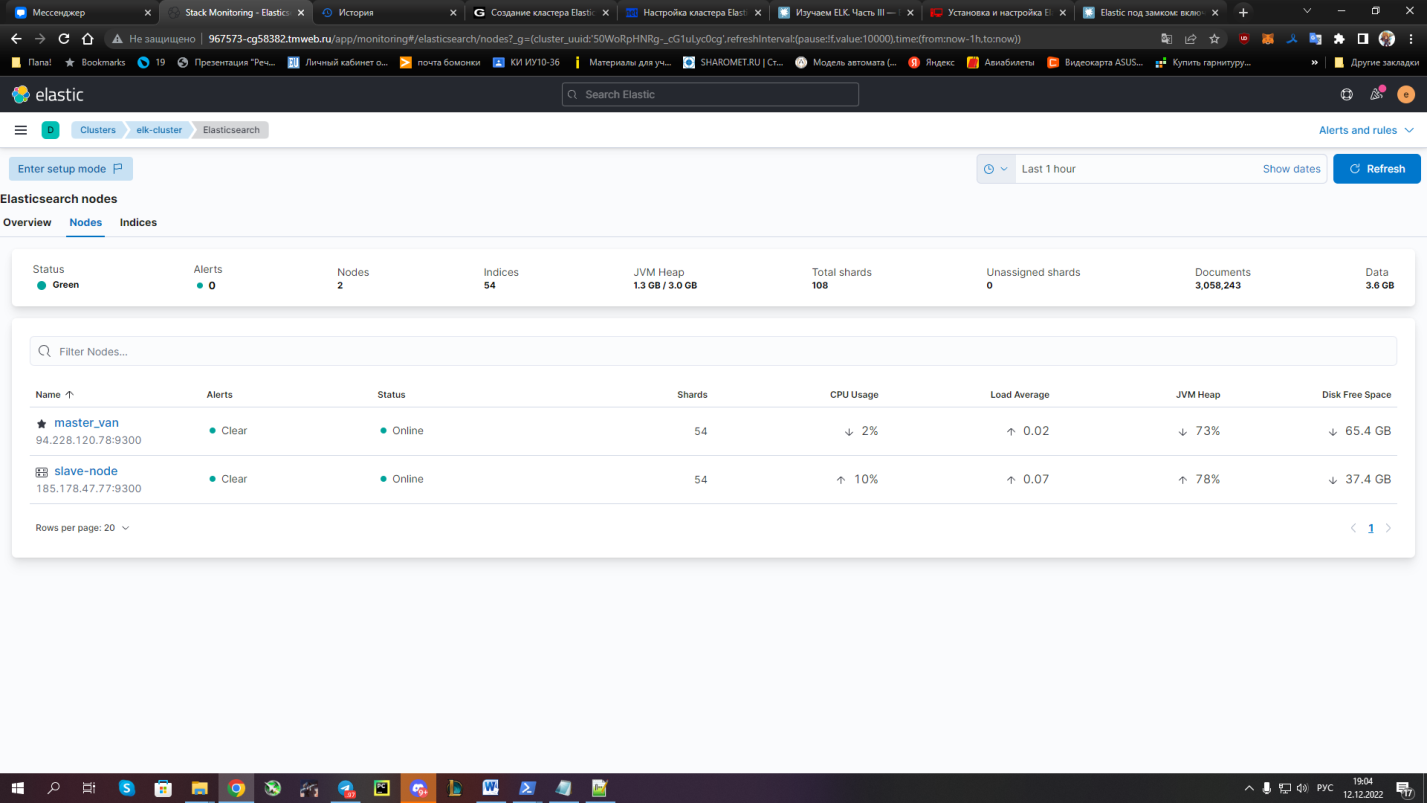
Проверяем, что всё распаковалось и появилось.





Также необходимо добавить в конфигурацию второй ноды наши сертификаты.





Далее перезапускаем Elasticsearch, заходим в Kibana и видим, что у нас добавилась 2 нода в кластере.

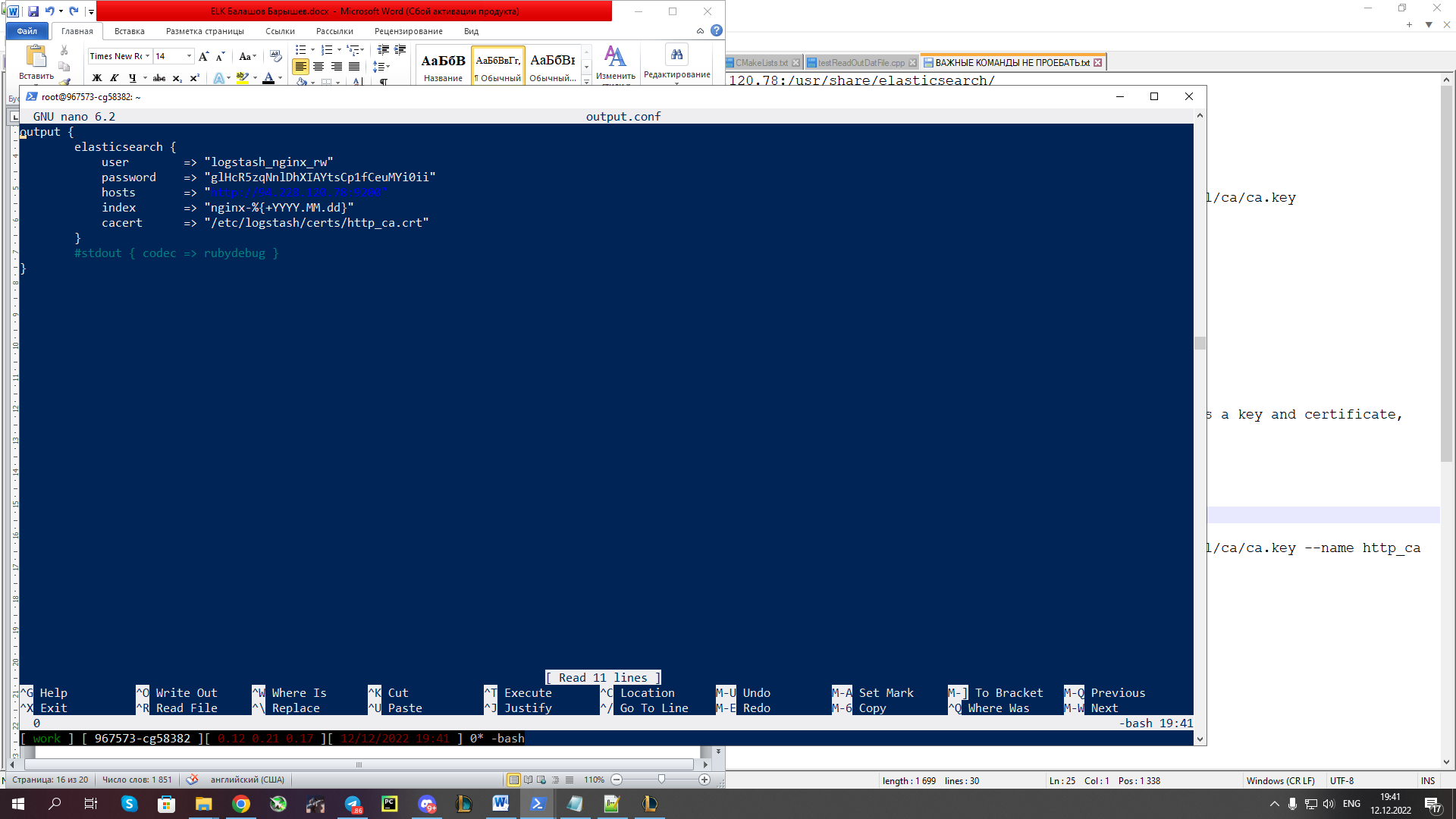
Обнаруживаем также, что у нас почему-то пропали логи из Logstash.

Для этого нам нужно добавить сертификаты и ключи для Logstash следующей командой:

**openssl pkcs8 -inform PEM -in /etc/logstash/certs/http\_ca.key -topk8 -nocrypt -outform PEM –out**

**/etc/logstash/certs/http\_ca.pkcs8.key**

Далее необходимо зайти в **output.conf** и добавить директорию с нашими сертификатами в строку **cacert:**



После сохранения конфигурации перезапускаем Logstash и Elasticsearch.  
Заходим в веб-интерфейс Kibana и видим, что логи снова поступают.

**Заключение**

В результате проделанной работы по созданию ELK кластера нам удалось поднять и запустить кластер, состоящий из двух нод и лично познакомиться со всеми тонкостями настройки узлов кластера, а также исправить все возникшие по ходу работы ошибки.

Сегодня наиболее популярным инструментом для построения решений по мониторингу корпоративной инфраструктуры считается именно ELK.

Мы выяснили, какое практическое применение может быть у нашего проекта, а именно с помощью ELK можно решать такие задачи как:

1. Агрегация товаров с разных интернет-магазинов и выполнение поиска по разным свойствам товара.
2. Организация централизованной системы хранения и анализа лог-файлов систем.
3. Агрегация данных различных систем и формирование показателей для аналитики состояния бизнес-процессов.
4. Работа с кулинарными книгами, построение различной отчетности в системах общепита.
5. Организация систем анкетирования и опросов.
6. Просмотр и анализ всевозможной неструктурированной статистической информации.

Также, отдельным пунктом хочется выделить совместимость ELK с

системой безопасности SIEM (Security information and event management).

Эта система с помощью ELK может осуществлять:

* Сбор, обработку и анализ событий безопасности, поступающих в систему из множества источников.
* Обнаружение в режиме реального времени атак и нарушений критериев и [политик безопасности](https://ru.wikipedia.org/wiki/%D0%9F%D0%BE%D0%BB%D0%B8%D1%82%D0%B8%D0%BA%D0%B0_%D0%B1%D0%B5%D0%B7%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D0%B8).
* Оперативно оценивать защищенность критически важных информационных ресурсов.