1. List some logging and visualization tools available in the market with the preferred scenario to use one over other.
   Answer:
   Some of the logging and visualization tools available in the market are:
   - **ELK Stack**
     "ELK" is the acronym for three open source projects: Elasticsearch, Logstash, and Kibana. Elasticsearch is a search and analytics engine. Logstash is a server‑side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a "stash" like Elasticsearch. Kibana lets users visualize data with charts and graphs in Elasticsearch.

   - **LOGalyze**
     LOGalyze is an open source, centralized log management and network monitoring software. If you would like to handle all of your log data in one place, LOGalyze is the right choice. It supports Linux/Unix servers, network devices, and Windows hosts. It provides real-time event detection and extensive search capabilities. With this open source application log analyzer, collect your log data from any device, analyze, normalize and parse them with any custom made Log Definition, use the built-in Statistics and Report Definitions or use your own ones.

   - **Datadog**
     Datadog is an observability service for cloud-scale applications, providing monitoring of servers, databases, tools, and services, through a SaaS-based data analytics platform.

   - **Splunk**
     Splunk helps capture, index and correlate real-time data in a searchable repository, from which it can generate graphs, reports, alerts, dashboards and visualizations. Splunk uses machine data for identifying data patterns, providing metrics, diagnosing problems and providing intelligence for business operations. Splunk is a horizontal technology used for application management, security and compliance, as well as business and web analytics.

   - **SolarWinds Papertrail**
     SolarWinds Papertrail is simple, powerful log management designed by engineers, for engineers, that supports most every log type, and provides

real-time log tailing and intuitive search and filter capabilities to simplify troubleshooting and reduce Mean Time to Repair (MTTR).

- **LogDNA**
  LogDNA is a highly scalable log management solution that indexes, aggregates, and analyzes log data. It supports logs originating from any source including applications, cloud systems, orchestration platforms like Kubernetes, bare metal servers, and more. We are built for any infrastructure, any architecture, any language. Its modular architecture allows it to grow to handle any log volume, whether you're logging 100 GB of data or 100 TB per day.

- **Graylog**
  Graylog is a leading centralized log management solution built to open standards for capturing, storing, and enabling real-time analysis of terabytes of machine data. We deliver a better user experience by making analysis ridiculously fast and efficient using a more cost-effective and flexible architecture.

- **Logfusion**
  LogFusion is a powerful real time log monitoring application designed for system administrators and developers! Use custom highlighting rules, filtering and more. You can even sync your LogFusion settings between computers.

- **XpoLog**
  XpoLog Center is an End-To-End solution for log management and log analysis. Every single computing device and application in your organization generates log events. Some of the most critical and private information in your organization is stored in the endless haystacks of logged data.

- **Google Cloud Logging**
  Cloud Logging empowers customers to manage, analyze, monitor, and gain insights from log data in real time.

2. Mention 10 best practises when logging. Why is log formatting necessary?
   Answer:
   The 10 best practises when logging are:
   I. **Know What Logs to Monitor, and What Not to Monitor**
      Know what not to log. Just because you can log something doesn't mean you should — and logging too much data can make it harder to find the data that actually matters. It also adds complexity to your log storage and management processes because it gives you more logs to manage.

   II. **Implement a Log Security and Retention Policy**
      Logs contain sensitive data. A log security policy should review sensitive data – like personal data of your clients or internal access keys for APIs. Make sure that sensitive data gets anonymized or encrypted before you ship logs to any third party. GDPR log management best practices teach you about good practices for data protection of sensitive data and personal data in web server logs. The secure transport of log data to log management servers requires the setup of encrypted endpoints for TLS or HTTPS on client and server side.

   III. **Design a Scalable and Reliable Log Storage**
      Planning the capacity for log storage should consider high load peaks. When systems run well, the amount of data produced per day is nearly constant and depends mainly on the system utilization and amount of transactions per day. In the case of critical system errors, we typically see accelerated growth in the log volume. If the log storage hits storage limits, you lose the latest logs, which are essential to fix system errors. The log storage must work as a cyclic buffer, which deletes the oldest data first before any storage limit is applied.

   IV. **Use the Proper Log Level**
      Among others, one important piece of information each log entry should include is the appropriate log level since it helps differentiate severe and important events from irregular or even regular events. Log levels will depend on the framework, but ERROR, WARNING, INFO, DEBUG are usually available. You need to be consistent in assigning severity to log messages. Overthewise filtering by severity will not work as expected.

   V. **Create Meaningful Log Messages**
      Readable and useful log messages are key for faster troubleshooting. If logs contain only some error codes or 'cryptic' error messages it can be

difficult to understand. As a Developer, you can save your organization a lot of time by providing a meaningful log message.

VI. **Use Structured Log Formats**
The log format should be structured (e.g., JSON or key/value format) having various fields like timestamp, severity, message and any other relevant data fields like process ID, transaction ID, etc. If you don't use a unique log format for all your applications, normalize the logs in the log shipper. Parse logs and store logs in a structured format.

VII. **Make Log Level Configurable**
Some application logs are too verbose and other application logs don't provide enough information about the activities. Adjustable log levels are the key to configure the verbosity of logs. Another topic for log reviews is the challenge to balance between logging relevant information and not exposing personal data or security-related information. If so, make sure that those messages can be anonymized or encrypted.

VIII. **Inspect Audit Logs Frequently**
Acting on security issues is crucial – so you should always have an eye on audit logs. Setup security tools such as auditd or OSSEC agents. The tools implement real-time log analysis and generate alert logs pointing to potential security issues. On top of such audit logs, you should define alerts on logs in order to be notified quickly of any suspicious activity. For more details, check out a quick tutorial on using auditd, plus you'll find some complementary frameworks too.

IX. **Review & Maintain Your Logs Constantly**
Unmaintained log data could lead to longer troubleshooting times, risks of exposing sensitive data or higher costs for log storage. Review the log output of your applications and adjust it to your needs. Reviews should cover usability, operational, and security aspects.

X. **View Logging as an Enabler of GitOps**
Connect the dots. Logging is one part of an entire monitoring strategy. To practice truly effective monitoring and alerting, you need to complement your logging with other types of monitoring like monitoring based on events, alerts and tracing. This is the only way to get the whole story of what's happening at any point in time. Logs are great for giving you high-definition details on issues, but this is useful only once you've seen

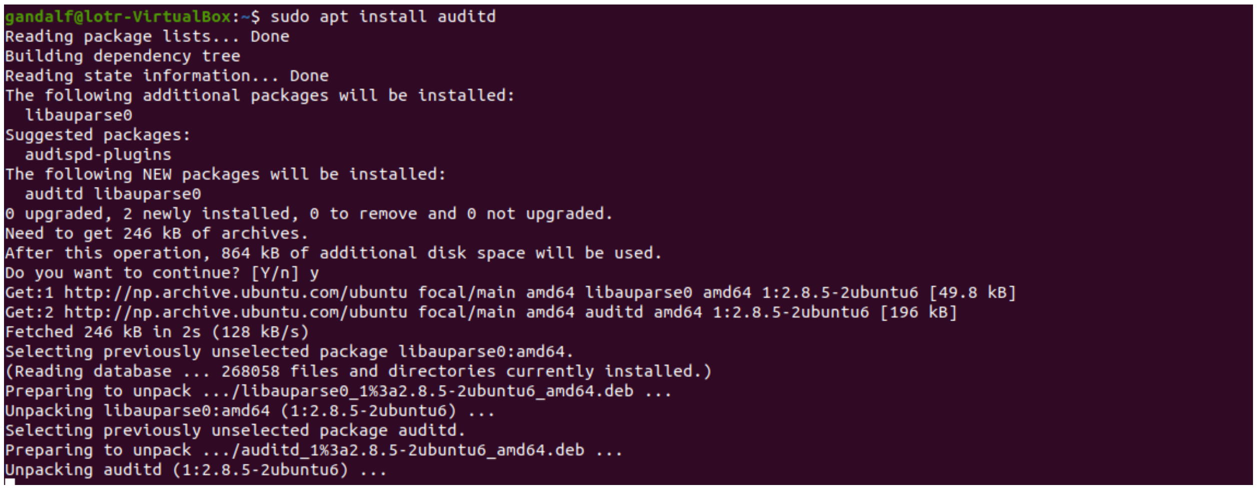
the forest and are ready to zoom into the trees. Metrics and events at an aggregate level may be more effective, especially when starting to troubleshoot an issue.

When it comes to creating readable log files, log formatting is key. The fundamental issue with log files, and the necessity for a structured format, is that they are often unstructured text data, making it impossible to extract usable information from them. A log format is a structured format for making logs machine-readable and parsable. This is the power of structured logs and a log management system that can handle them. One of the most important characteristics of log management software is the ability to convert raw data into something that is immediately understandable and easy to read.

3. Create a file in your system. Whenever someone performs some action(read, write, execute) on that file, the event should be logged somewhere.

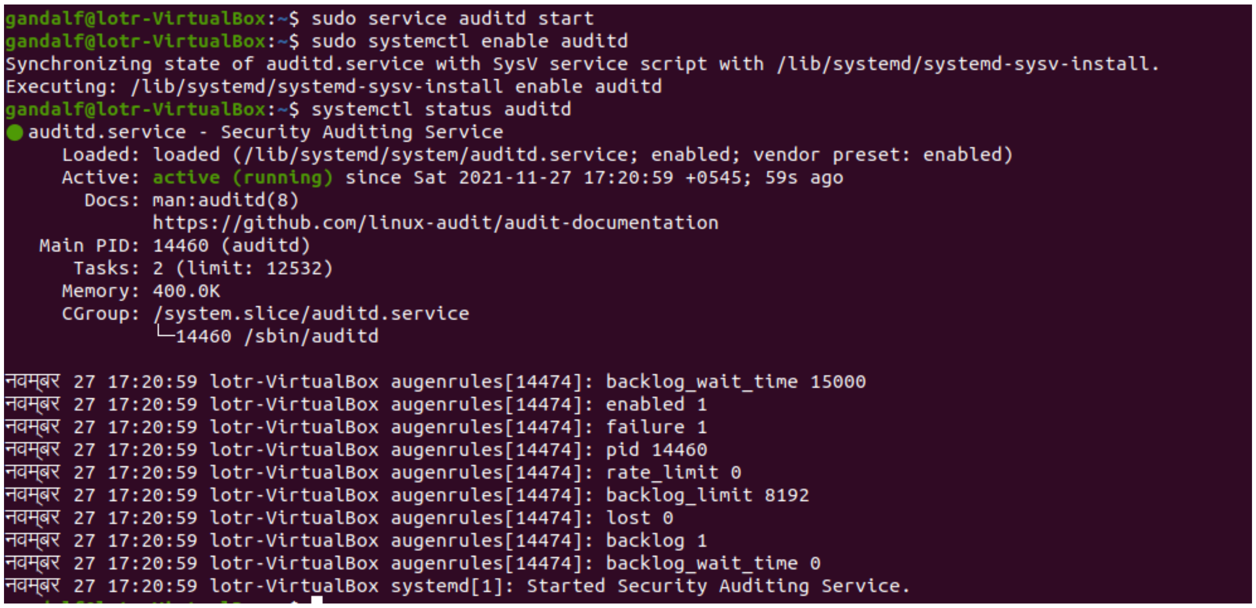
   The Linux Auditing System helps system administrators create an audit trail, a log for every action on the server. We can track security-relevant events, record the events in a log file, and detect misuse or unauthorized activities by inspecting the audit log files. We can choose which actions on the server to monitor and to what extent. Audit does not provide additional security to your system, rather, it helps track any violations of system policies and enables you to take additional security measures to prevent them.

   Firstly let's install Aduit using the following command:
   - *sudo apt install auditd*



```
gandalf@lotr-VirtualBox:~$ sudo apt install auditd
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libauparse0
Suggested packages:
  audispd-plugins
The following NEW packages will be installed:
  auditd libauparse0
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 246 kB of archives.
After this operation, 864 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://np.archive.ubuntu.com/ubuntu focal/main amd64 libauparse0 amd64 1:2.8.5-2ubuntu6 [49.8 kB]
Get:2 http://np.archive.ubuntu.com/ubuntu focal/main amd64 auditd amd64 1:2.8.5-2ubuntu6 [196 kB]
Fetched 246 kB in 2s (128 kB/s)
Selecting previously unselected package libauparse0:amd64.
(Reading database ... 268058 files and directories currently installed.)
Preparing to unpack .../libauparse0_1%3a2.8.5-2ubuntu6_amd64.deb ...
Unpacking libauparse0:amd64 (1:2.8.5-2ubuntu6) ...
Selecting previously unselected package auditd.
Preparing to unpack .../auditd_1%3a2.8.5-2ubuntu6_amd64.deb ...
Unpacking auditd (1:2.8.5-2ubuntu6) ...
```

- Now let’s start, enable and check the status of auditd, we just installed using the following commands:
    - *sudo service auditd start*
    - *sudo systemctl enable auditd*
    - *systemctl status auditd*

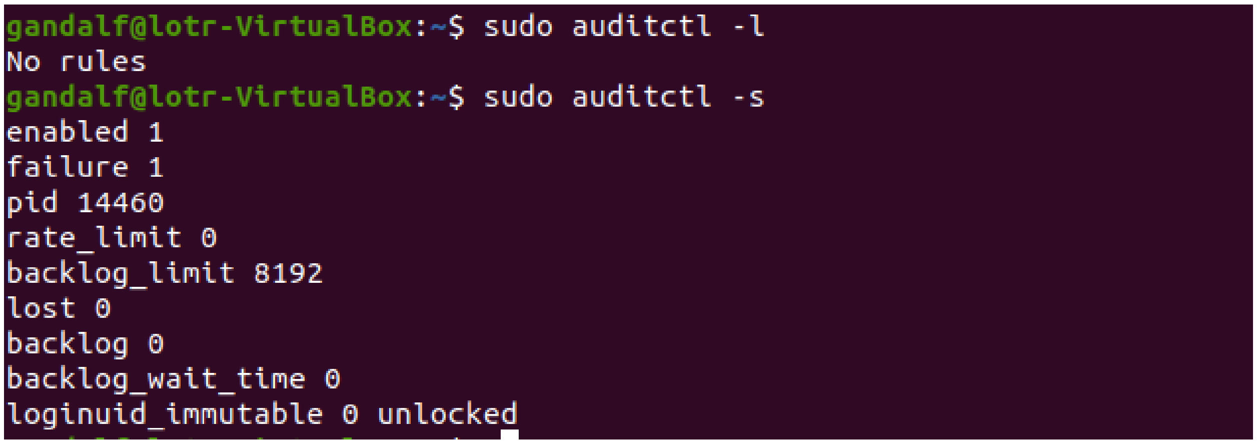


```
gandalf@lotr-VirtualBox:~$ sudo service auditd start
gandalf@lotr-VirtualBox:~$ sudo systemctl enable auditd
Synchronizing state of auditd.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable auditd
gandalf@lotr-VirtualBox:~$ systemctl status auditd
● auditd.service - Security Auditing Service
     Loaded: loaded (/lib/systemd/system/auditd.service; enabled; vendor preset: enabled)
     Active: active (running) since Sat 2021-11-27 17:20:59 +0545; 59s ago
       Docs: man:auditd(8)
             https://github.com/linux-audit/audit-documentation
   Main PID: 14460 (auditd)
      Tasks: 2 (limit: 12532)
     Memory: 400.0K
     CGroup: /system.slice/auditd.service
             └─14460 /sbin/auditd

नवम्बर 27 17:20:59 lotr-VirtualBox augenrules[14474]: backlog_wait_time 15000
नवम्बर 27 17:20:59 lotr-VirtualBox augenrules[14474]: enabled 1
नवम्बर 27 17:20:59 lotr-VirtualBox augenrules[14474]: failure 1
नवम्बर 27 17:20:59 lotr-VirtualBox augenrules[14474]: pid 14460
नवम्बर 27 17:20:59 lotr-VirtualBox augenrules[14474]: rate_limit 0
नवम्बर 27 17:20:59 lotr-VirtualBox augenrules[14474]: backlog_limit 8192
नवम्बर 27 17:20:59 lotr-VirtualBox augenrules[14474]: lost 0
नवम्बर 27 17:20:59 lotr-VirtualBox augenrules[14474]: backlog 1
नवम्बर 27 17:20:59 lotr-VirtualBox augenrules[14474]: backlog_wait_time 0
नवम्बर 27 17:20:59 lotr-VirtualBox systemd[1]: Started Security Auditing Service.
```

Now let’s view the audit rules:

- We can view the current set of rules using the command
    - *sudo auditctl -l*
    - It will show no rules, as we have not configured anything and it's the default.
- And the current status of the audit system can be viewed using:
    - *sudo auditctl -s*



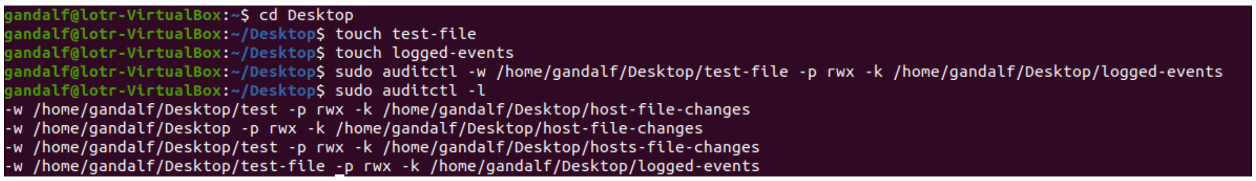
```
gandalf@lotr-VirtualBox:~$ sudo auditctl -l
No rules
gandalf@lotr-VirtualBox:~$ sudo auditctl -s
enabled 1
failure 1
pid 14460
rate_limit 0
backlog_limit 8192
lost 0
backlog 0
backlog_wait_time 0
loginuid_immutable 0 unlocked
```

There are 3 kinds of audit rules:

- **Control rules:** These rules are used for changing the configuration and settings of the audit system itself.
- **Filesystem rules:** These are file or directory watches. Using these rules, we can audit any kind of access to specific files or directories.
- **System call rules:** These rules are used for monitoring system calls made by any process or a particular user.

For this task we will be configuring FILE SYSTEM RULES:

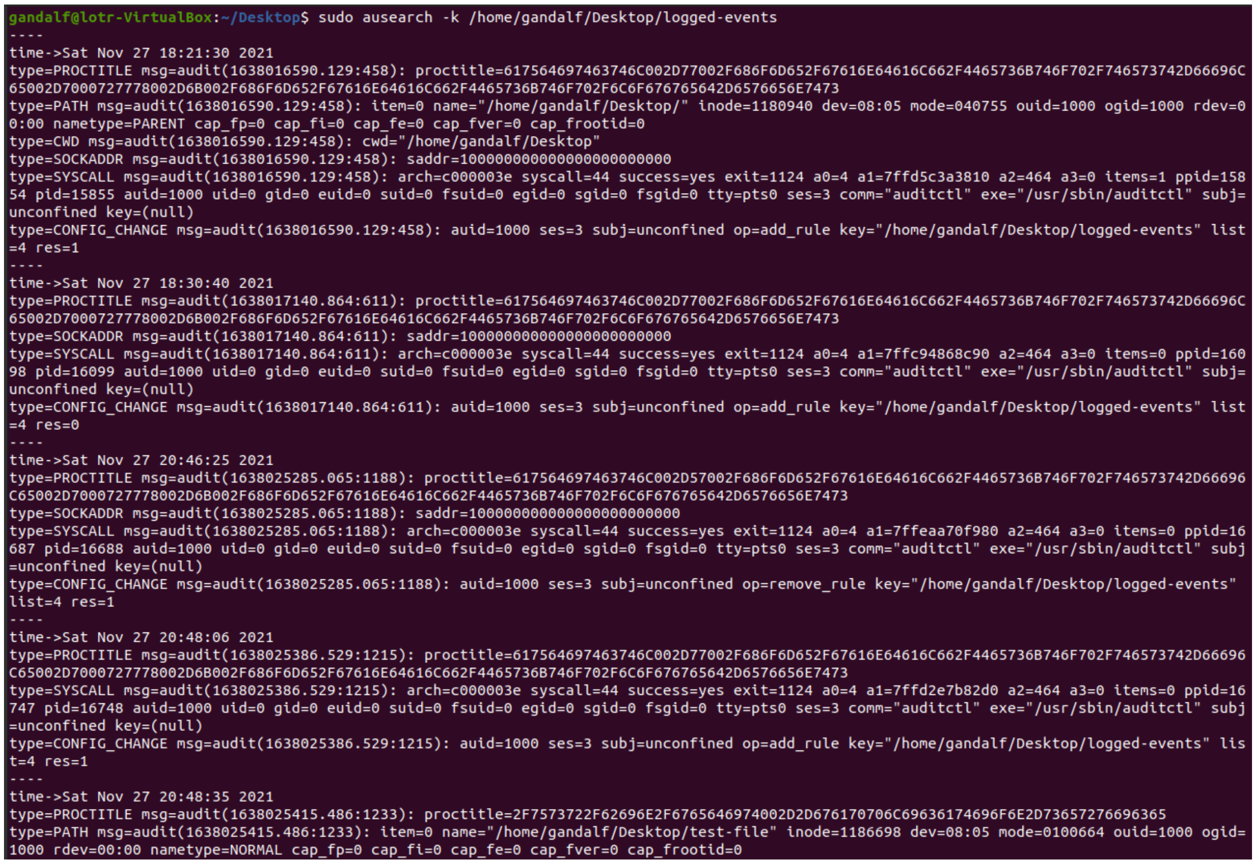
- Filesystem watches can be set on files and directories. We can specify the type of access to watch. The syntax for a filesystem rule is:
    - *sudo auditctl -w <path-to-file> -p <permissions> -k <key-name>*
    - Where,
        - Path-to-file: is the file or directory that is audited.
        - Permissions: the permissions that are logged, This value can be one or a combination of r(read), w(write), x(execute), and a(attribute change).
        - Key-name: this is an optional string that helps you identify which rule(s) generated a particular log entry.
    - In our case let's make a test file in our Desktop and name it 'test' and let's log the events in this file using following command on Desktop:
        - *touch test-file*
    - And let's create a file named 'hosts-file-changes' again on the Desktop and here we will record the logs. LEt's use the following commands for this:
        - *touch logged-events*
    - Now let's use this command to set a rule which will ask the audit system to watch for read access, write access, and attribute access changes to the file on '*/home/gandalf/Desktop/test*'.
        - *sudo auditctl -w /home/gandalf/Desktop/test-file -p rwx -k /home/gandalf/Desktop/logged-events*
    - We can use the following command to view the rules we just added and we can then verify:
        - *sudo auditctl -l*



```
gandalf@lotr-VirtualBox:~$ cd Desktop
gandalf@lotr-VirtualBox:~/Desktop$ touch test-file
gandalf@lotr-VirtualBox:~/Desktop$ touch logged-events
gandalf@lotr-VirtualBox:~/Desktop$ sudo auditctl -w /home/gandalf/Desktop/test-file -p rwx -k /home/gandalf/Desktop/logged-events
gandalf@lotr-VirtualBox:~/Desktop$ sudo auditctl -l
-w /home/gandalf/Desktop/test -p rwx -k /home/gandalf/Desktop/host-file-changes
-w /home/gandalf/Desktop -p rwx -k /home/gandalf/Desktop/host-file-changes
-w /home/gandalf/Desktop/test -p rwx -k /home/gandalf/Desktop/hosts-file-changes
-w /home/gandalf/Desktop/test-file -p rwx -k /home/gandalf/Desktop/logged-events
```

        - I have more than one rule set because the others were set when I was practising for this task.

- Now let’s make some manual changes to the test file so as to check if we have done everything right and everything is up and running.
- After all this let’s search the audit logs for the events, for this we can use the command *ausearch* i.e:
    - *sudo ausearch -k <-key-name>*
    - For our case the command will be:
        - *sudo ausearch -k /home/gandalf/Desktop/logged-events*

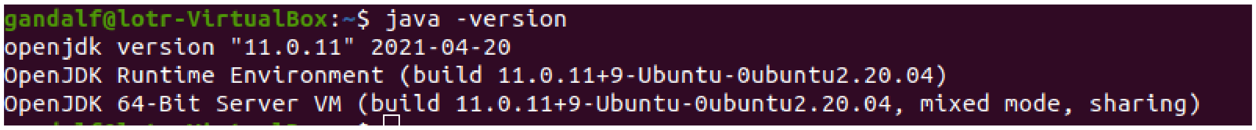


```
gandalf@lotr-VirtualBox:~/Desktop$ sudo ausearch -k /home/gandalf/Desktop/logged-events
----
time->Sat Nov 27 18:21:30 2021
type=PROCTITLE msg=audit(1638016590.129:458): proctitle=617564697463746C002D77002F686F6D652F67616E64616C662F4465736B746F702F746573742D66696C
65002D7000727778002D6B002F686F6D652F67616E64616C662F4465736B746F702F6C6F676765642D6576656E7473
type=PATH msg=audit(1638016590.129:458): item=0 name="/home/gandalf/Desktop/" inode=1180940 dev=08:05 mode=040755 ouid=1000 ogid=1000 rdev=0
0:00 nametype=PARENT cap_fp=0 cap_fi=0 cap_fe=0 cap_fver=0 cap_frootid=0
type=CWD msg=audit(1638016590.129:458): cwd="/home/gandalf/Desktop"
type=SOCKADDR msg=audit(1638016590.129:458): saddr=100000000000000000000000
type=SYSCALL msg=audit(1638016590.129:458): arch=c000003e syscall=44 success=yes exit=1124 a0=4 a1=7ffd5c3a3810 a2=464 a3=0 items=1 ppid=158
54 pid=15855 auid=1000 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=3 comm="auditctl" exe="/usr/sbin/auditctl" subj=
unconfined key=(null)
type=CONFIG_CHANGE msg=audit(1638016590.129:458): auid=1000 ses=3 subj=unconfined op=add_rule key="/home/gandalf/Desktop/logged-events" list
=4 res=1
----
time->Sat Nov 27 18:30:40 2021
type=PROCTITLE msg=audit(1638017140.864:611): proctitle=617564697463746C002D77002F686F6D652F67616E64616C662F4465736B746F702F746573742D66696C
65002D7000727778002D6B002F686F6D652F67616E64616C662F4465736B746F702F6C6F676765642D6576656E7473
type=SOCKADDR msg=audit(1638017140.864:611): saddr=100000000000000000000000
type=SYSCALL msg=audit(1638017140.864:611): arch=c000003e syscall=44 success=yes exit=1124 a0=4 a1=7ffc94868c90 a2=464 a3=0 items=0 ppid=160
98 pid=16099 auid=1000 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=3 comm="auditctl" exe="/usr/sbin/auditctl" subj=
unconfined key=(null)
type=CONFIG_CHANGE msg=audit(1638017140.864:611): auid=1000 ses=3 subj=unconfined op=add_rule key="/home/gandalf/Desktop/logged-events" list
=4 res=0
----
time->Sat Nov 27 20:46:25 2021
type=PROCTITLE msg=audit(1638025285.065:1188): proctitle=617564697463746C002D57002F686F6D652F67616E64616C662F4465736B746F702F746573742D66696
C65002D7000727778002D6B002F686F6D652F67616E64616C662F4465736B746F702F6C6F676765642D6576656E7473
type=SOCKADDR msg=audit(1638025285.065:1188): saddr=100000000000000000000000
type=SYSCALL msg=audit(1638025285.065:1188): arch=c000003e syscall=44 success=yes exit=1124 a0=4 a1=7ffeaa70f980 a2=464 a3=0 items=0 ppid=16
687 pid=16688 auid=1000 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=3 comm="auditctl" exe="/usr/sbin/auditctl" subj
=unconfined key=(null)
type=CONFIG_CHANGE msg=audit(1638025285.065:1188): auid=1000 ses=3 subj=unconfined op=remove_rule key="/home/gandalf/Desktop/logged-events"
list=4 res=1
----
time->Sat Nov 27 20:48:06 2021
type=PROCTITLE msg=audit(1638025386.529:1215): proctitle=617564697463746C002D77002F686F6D652F67616E64616C662F4465736B746F702F746573742D66696
C65002D7000727778002D6B002F686F6D652F67616E64616C662F4465736B746F702F6C6F676765642D6576656E7473
type=SYSCALL msg=audit(1638025386.529:1215): arch=c000003e syscall=44 success=yes exit=1124 a0=4 a1=7ffd2e7b82d0 a2=464 a3=0 items=0 ppid=16
747 pid=16748 auid=1000 uid=0 gid=0 euid=0 suid=0 fsuid=0 egid=0 sgid=0 fsgid=0 tty=pts0 ses=3 comm="auditctl" exe="/usr/sbin/auditctl" subj
=unconfined key=(null)
type=CONFIG_CHANGE msg=audit(1638025386.529:1215): auid=1000 ses=3 subj=unconfined op=add_rule key="/home/gandalf/Desktop/logged-events" lis
t=4 res=1
----
time->Sat Nov 27 20:48:35 2021
type=PROCTITLE msg=audit(1638025415.486:1233): proctitle=2F7573722F62696E2F6765646974002D2D676170706C69636174696F6E2D73657276696365
type=PATH msg=audit(1638025415.486:1233): item=0 name="/home/gandalf/Desktop/test-file" inode=1186698 dev=08:05 mode=0100664 ouid=1000 ogid=
1000 rdev=00:00 nametype=NORMAL cap_fp=0 cap_fi=0 cap_fe=0 cap_fver=0 cap_frootid=0
```

We can observe that the events have been recorded and we have successfully set up a file in our system in which all the events of the action of read, write and execute will be logged.

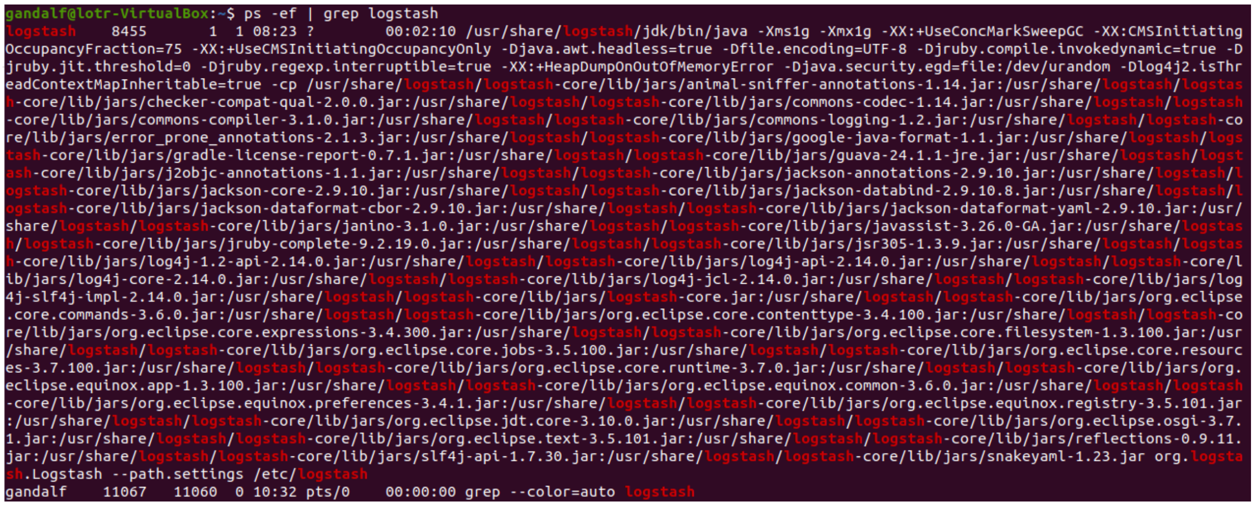
4. Install logstash in your system. download a sample nginx log from https://github.com/elastic/examples/blob/master/Common%20Data%20Formats/nginx_logs/nginx_logs , parse the logs using logstash. The parsed output must contain the geographical information like country, state etc. that the request is originating from. save the parsed output to a file in your system.

   Logstash is a log-parsing software used to collect logs and store them on Elasticsearch.
   To install Logstash on our system, We need to follow the steps given below:
   - Firstly we need to have Java installed in our system. Then only we need to proceed to the next step. We can verify the java installation using the following command on our linux machine:
     - *java -version*



```
gandalf@lotr-VirtualBox:~$ java -version
openjdk version "11.0.11" 2021-04-20
OpenJDK Runtime Environment (build 11.0.11+9-Ubuntu-0ubuntu2.20.04)
OpenJDK 64-Bit Server VM (build 11.0.11+9-Ubuntu-0ubuntu2.20.04, mixed mode, sharing)
```

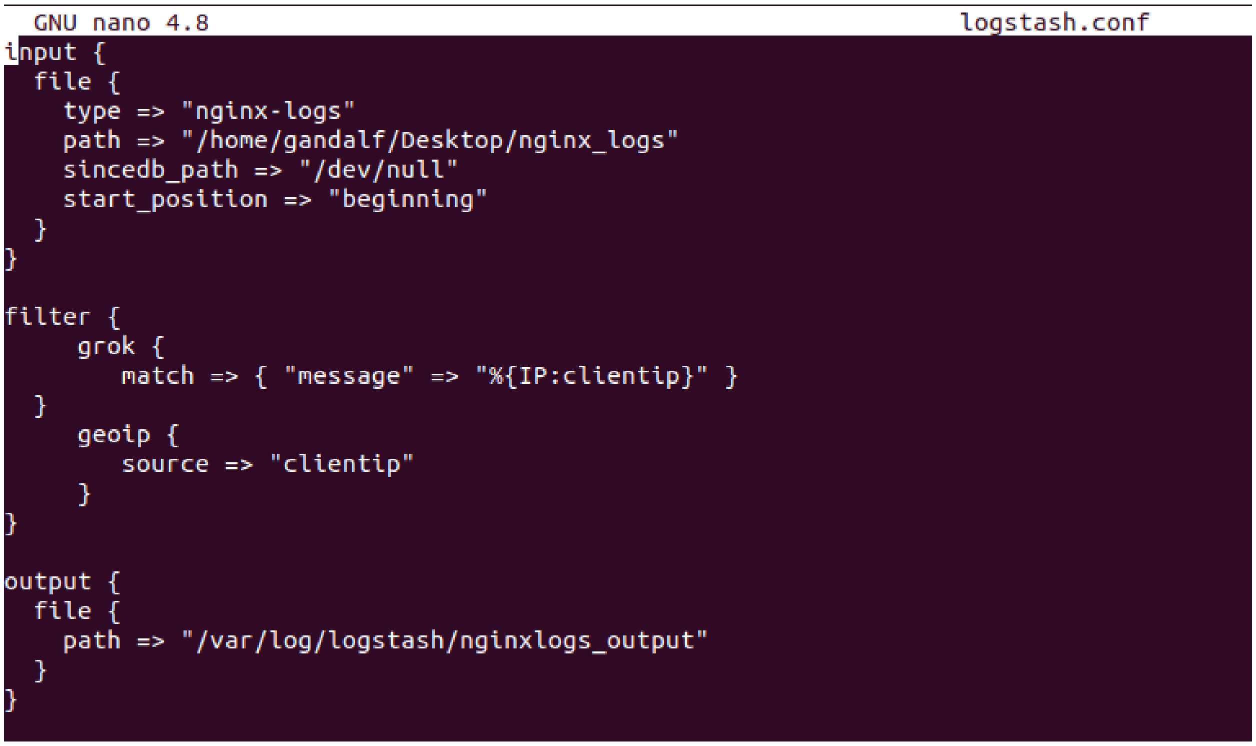
   - Now we can install Logstash easily with the following command:
     - *sudo apt-get install logstash -y*
   - Once the Logstash is installed, we need to enable it to start at system reboot using the following commands:
     - *systemctl start logstash*
     - *systemctl enable logstash*
   - We can check if our logstash is running on our system by using following command:
     - *ps -ef | grep logstash*



```
gandalf@lotr-VirtualBox:~$ ps -ef | grep logstash
logstash    8455       1  1 08:23 ?        00:02:10 /usr/share/logstash/jdk/bin/java -Xms1g -Xmx1g -XX:+UseConcMarkSweepGC -XX:CMSInitiating
OccupancyFraction=75 -XX:+UseCMSInitiatingOccupancyOnly -Djava.awt.headless=true -Dfile.encoding=UTF-8 -Djruby.compile.invokedynamic=true -D
jruby.jit.threshold=0 -Djruby.regexp.interruptible=true -XX:+HeapDumpOnOutOfMemoryError -Djava.security.egd=file:/dev/urandom -Dlog4j2.isThr
eadContextMapInheritable=true -cp /usr/share/logstash/logstash-core/lib/jars/animal-sniffer-annotations-1.14.jar:/usr/share/logstash/logstas
h-core/lib/jars/checker-compat-qual-2.0.0.jar:/usr/share/logstash/logstash-core/lib/jars/commons-codec-1.14.jar:/usr/share/logstash/logstash
-core/lib/jars/commons-compiler-3.1.0.jar:/usr/share/logstash/logstash-core/lib/jars/commons-logging-1.2.jar:/usr/share/logstash/logstash-co
re/lib/jars/error_prone_annotations-2.1.3.jar:/usr/share/logstash/logstash-core/lib/jars/google-java-format-1.1.jar:/usr/share/logstash/logs
tash-core/lib/jars/gradle-license-report-0.7.1.jar:/usr/share/logstash/logstash-core/lib/jars/guava-24.1.1-jre.jar:/usr/share/logstash/logst
ash-core/lib/jars/j2objc-annotations-1.1.jar:/usr/share/logstash/logstash-core/lib/jars/jackson-annotations-2.9.10.jar:/usr/share/logstash/l
ogstash-core/lib/jars/jackson-core-2.9.10.jar:/usr/share/logstash/logstash-core/lib/jars/jackson-databind-2.9.10.8.jar:/usr/share/logstash/l
ogstash-core/lib/jars/jackson-dataformat-cbor-2.9.10.jar:/usr/share/logstash/logstash-core/lib/jars/jackson-dataformat-yaml-2.9.10.jar:/usr/
share/logstash/logstash-core/lib/jars/janino-3.1.0.jar:/usr/share/logstash/logstash-core/lib/jars/javassist-3.26.0-GA.jar:/usr/share/logstas
h/logstash-core/lib/jars/jruby-complete-9.2.19.0.jar:/usr/share/logstash/logstash-core/lib/jars/jsr305-1.3.9.jar:/usr/share/logstash/logstas
h-core/lib/jars/log4j-1.2-api-2.14.0.jar:/usr/share/logstash/logstash-core/lib/jars/log4j-api-2.14.0.jar:/usr/share/logstash/logstash-core/l
ib/jars/log4j-core-2.14.0.jar:/usr/share/logstash/logstash-core/lib/jars/log4j-jcl-2.14.0.jar:/usr/share/logstash/logstash-core/lib/jars/log
4j-slf4j-impl-2.14.0.jar:/usr/share/logstash/logstash-core/lib/jars/logstash-core.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse
.core.commands-3.6.0.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.core.contenttype-3.4.100.jar:/usr/share/logstash/logstash-co
re/lib/jars/org.eclipse.core.expressions-3.4.300.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.core.filesystem-1.3.100.jar:/usr
/share/logstash/logstash-core/lib/jars/org.eclipse.core.jobs-3.5.100.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.core.resourc
es-3.7.100.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.core.runtime-3.7.0.jar:/usr/share/logstash/logstash-core/lib/jars/org.
eclipse.equinox.app-1.3.100.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.equinox.common-3.6.0.jar:/usr/share/logstash/logstash
-core/lib/jars/org.eclipse.equinox.preferences-3.4.1.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.equinox.registry-3.5.101.jar
:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.jdt.core-3.10.0.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.osgi-3.7.
1.jar:/usr/share/logstash/logstash-core/lib/jars/org.eclipse.text-3.5.101.jar:/usr/share/logstash/logstash-core/lib/jars/reflections-0.9.11.
jar:/usr/share/logstash/logstash-core/lib/jars/slf4j-api-1.7.30.jar:/usr/share/logstash/logstash-core/lib/jars/snakeyaml-1.23.jar org.logsta
sh.Logstash --path.settings /etc/logstash
gandalf    11067   11060  0 10:32 pts/0    00:00:00 grep --color=auto logstash
```

   - Now let's need to configure the input, filter, and the output plugins inside */etc/logstash/conf.d/* directory so as to parse our nginx log file producing the output with the geographical information like country, state etc using the following command:

- ○ *nano /etc/logstash/conf.d/logstash.conf*



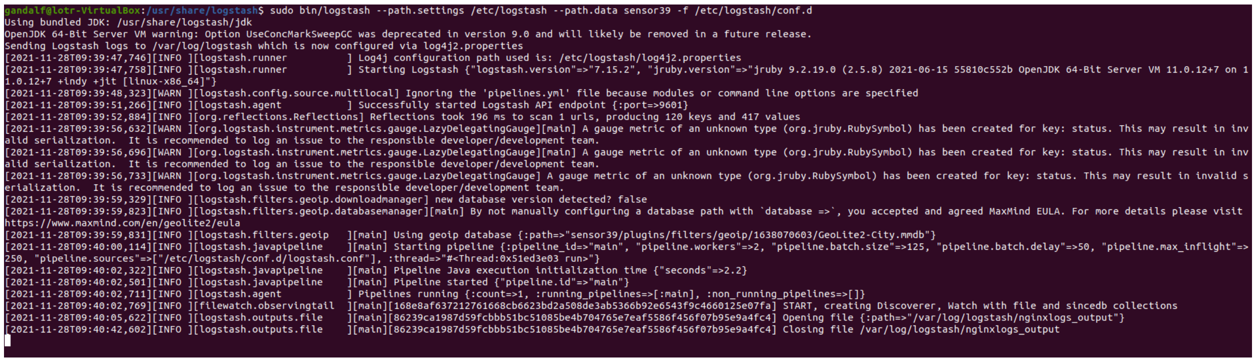
```
  GNU nano 4.8                                          logstash.conf
input {
  file {
    type => "nginx-logs"
    path => "/home/gandalf/Desktop/nginx_logs"
    sincedb_path => "/dev/null"
    start_position => "beginning"
  }
}

filter {
     grok {
        match => { "message" => "%{IP:clientip}" }
  }
     geoip {
        source => "clientip"
     }
}

output {
  file {
    path => "/var/log/logstash/nginxlogs_output"
  }
}
```

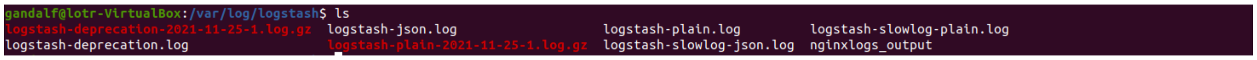
- ○ We can see the configuration file for our task in the snapshot above.

- After all the steps, let's navigate to the */usr/share/logstash* directory and run the following command so that the logstash will parse our nginx logs file as per configuration:
  - ○ *sudo bin/logstash --path.settings /etc/logstash --path.data sensor39 -f /etc/logstash/conf.d*



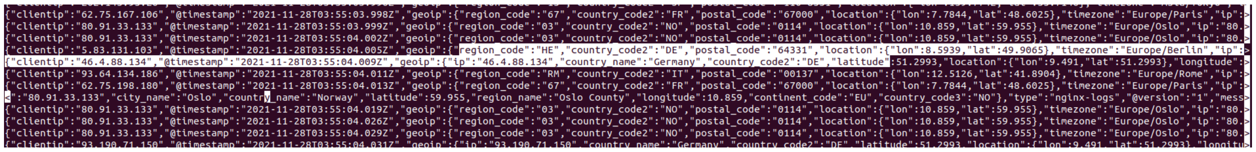
```
gandalf@lotr-VirtualBox:/usr/share/logstash$ sudo bin/logstash --path.settings /etc/logstash --path.data sensor39 -f /etc/logstash/conf.d
Using bundled JDK: /usr/share/logstash/jdk
OpenJDK 64-Bit Server VM warning: Option UseConcMarkSweepGC was deprecated in version 9.0 and will likely be removed in a future release.
Sending Logstash logs to /var/log/logstash which is now configured via log4j2.properties
[2021-11-28T09:39:47,746][INFO ][logstash.runner          ] Log4j configuration path used is: /etc/logstash/log4j2.properties
[2021-11-28T09:39:47,758][INFO ][logstash.runner          ] Starting Logstash {"logstash.version"=>"7.15.2", "jruby.version"=>"jruby 9.2.19.0 (2.5.8) 2021-06-15 55810c552b OpenJDK 64-Bit Server VM 11.0.12+7 on 1
1.0.12+7 +indy +jit [linux-x86_64]"}
[2021-11-28T09:39:48,323][WARN ][logstash.config.source.multilocal] Ignoring the 'pipelines.yml' file because modules or command line options are specified
[2021-11-28T09:39:51,266][INFO ][logstash.agent           ] Successfully started Logstash API endpoint {:port=>9601}
[2021-11-28T09:39:52,884][INFO ][org.reflections.Reflections] Reflections took 196 ms to scan 1 urls, producing 120 keys and 417 values
[2021-11-28T09:39:56,632][WARN ][org.logstash.instrument.metrics.gauge.LazyDelegatingGauge][main] A gauge metric of an unknown type (org.jruby.RubySymbol) has been created for key: status. This may result in inv
alid serialization.  It is recommended to log an issue to the responsible developer/development team.
[2021-11-28T09:39:56,696][WARN ][org.logstash.instrument.metrics.gauge.LazyDelegatingGauge][main] A gauge metric of an unknown type (org.jruby.RubySymbol) has been created for key: status. This may result in inv
alid serialization.  It is recommended to log an issue to the responsible developer/development team.
[2021-11-28T09:39:56,733][WARN ][org.logstash.instrument.metrics.gauge.LazyDelegatingGauge] A gauge metric of an unknown type (org.jruby.RubySymbol) has been created for key: status. This may result in invalid s
erialization.  It is recommended to log an issue to the responsible developer/development team.
[2021-11-28T09:39:59,329][INFO ][logstash.filters.geoip.downloadmanager] new database version detected? false
[2021-11-28T09:39:59,823][INFO ][logstash.filters.geoip.databasemanager][main] By not manually configuring a database path with `database =>`, you accepted and agreed MaxMind EULA. For more details please visit
https://www.maxmind.com/en/geolite2/eula
[2021-11-28T09:39:59,831][INFO ][logstash.filters.geoip   ][main] Using geoip database {:path=>"sensor39/plugins/filters/geoip/1638070603/GeoLite2-City.mmdb"}
[2021-11-28T09:40:00,114][INFO ][logstash.javapipeline    ][main] Starting pipeline {:pipeline_id=>"main", "pipeline.workers"=>2, "pipeline.batch.size"=>125, "pipeline.batch.delay"=>50, "pipeline.max_inflight"=>
250, "pipeline.sources"=>["/etc/logstash/conf.d/logstash.conf"], :thread=>"#<Thread:0x51ed3e03 run>"}
[2021-11-28T09:40:02,322][INFO ][logstash.javapipeline    ][main] Pipeline Java execution initialization time {"seconds"=>2.2}
[2021-11-28T09:40:02,501][INFO ][logstash.javapipeline    ][main] Pipeline started {"pipeline.id"=>"main"}
[2021-11-28T09:40:02,711][INFO ][logstash.agent           ] Pipelines running {:count=>1, :running_pipelines=>[:main], :non_running_pipelines=>[]}
[2021-11-28T09:40:02,769][INFO ][filewatch.observingtail  ][main][168e8af637212761668cb6623bd2a508de3ab5366b92e6543f9c4660125e07fa] START, creating Discoverer, Watch with file and sincedb collections
[2021-11-28T09:40:05,622][INFO ][logstash.outputs.file    ][main][86239ca1987d59fcbbb51bc51085be4b704765e7eaf5586f456f07b95e9a4fc4] Opening file {:path=>"/var/log/logstash/nginxlogs_output"}
[2021-11-28T09:40:42,602][INFO ][logstash.outputs.file    ][main][86239ca1987d59fcbbb51bc51085be4b704765e7eaf5586f456f07b95e9a4fc4] Closing file /var/log/logstash/nginxlogs_output
```

- After we execute the above command, we can check the logs out file in this path as we configured */var/log/logstash/nginxlogs_output.*



```
gandalf@lotr-VirtualBox:/var/log/logstash$ ls
logstash-deprecation-2021-11-25-1.log.gz  logstash-json.log               logstash-plain.log        logstash-slowlog-plain.log
logstash-deprecation.log                  logstash-plain-2021-11-25-1.log.gz  logstash-slowlog-json.log  nginxlogs_output
```

- We can see that the new nginxlogs_output file has been generated. This log output file has been push in this repository for the review also, and a small snapshot can be viewed below:



```
{"clientip":"62.75.167.106","@timestamp":"2021-11-28T03:55:03.998Z","geoip":{"region_code":"67","country_code2":"FR","postal_code":"67000","location":{"lon":7.7844,"lat":48.6025},"timezone":"Europe/Paris","ip":>
{"clientip":"80.91.33.133","@timestamp":"2021-11-28T03:55:03.999Z","geoip":{"region_code":"03","country_code2":"NO","postal_code":"0114","location":{"lon":10.859,"lat":59.955},"timezone":"Europe/Oslo","ip":"80.>
{"clientip":"80.91.33.133","@timestamp":"2021-11-28T03:55:04.002Z","geoip":{"region_code":"03","country_code2":"NO","postal_code":"0114","location":{"lon":10.859,"lat":59.955},"timezone":"Europe/Oslo","ip":"80.>
{"clientip":"5.83.131.103","@timestamp":"2021-11-28T03:55:04.005Z","geoip":{"region_code":"HE","country_code2":"DE","postal_code":"64331","location":{"lon":8.5939,"lat":49.9065},"timezone":"Europe/Berlin","ip":>
{"clientip":"46.4.88.134","@timestamp":"2021-11-28T03:55:04.009Z","geoip":{"ip":"46.4.88.134","country_name":"Germany","country_code2":"DE","latitude":51.2993,"location":{"lon":9.491,"lat":51.2993},"longitude":>
{"clientip":"93.64.134.186","@timestamp":"2021-11-28T03:55:04.011Z","geoip":{"region_code":"RM","country_code2":"IT","postal_code":"00137","location":{"lon":12.5126,"lat":41.8904},"timezone":"Europe/Rome","ip":>
{"clientip":"62.75.198.180","@timestamp":"2021-11-28T03:55:04.013Z","geoip":{"region_code":"67","country_code2":"FR","postal_code":"67000","location":{"lon":7.7844,"lat":48.6025},"timezone":"Europe/Paris","ip":>
<":"80.91.33.133","city_name":"Oslo","country_name":"Norway","latitude":59.955,"region_name":"Oslo County","longitude":10.859,"continent_code":"EU","country_code3":"NO"},"type":"nginx-logs","@version":"1","mess>
{"clientip":"80.91.33.133","@timestamp":"2021-11-28T03:55:04.019Z","geoip":{"region_code":"03","country_code2":"NO","postal_code":"0114","location":{"lon":10.859,"lat":59.955},"timezone":"Europe/Oslo","ip":"80.>
{"clientip":"80.91.33.133","@timestamp":"2021-11-28T03:55:04.026Z","geoip":{"region_code":"03","country_code2":"NO","postal_code":"0114","location":{"lon":10.859,"lat":59.955},"timezone":"Europe/Oslo","ip":"80.>
{"clientip":"80.91.33.133","@timestamp":"2021-11-28T03:55:04.029Z","geoip":{"region_code":"03","country_code2":"NO","postal_code":"0114","location":{"lon":10.859,"lat":59.955},"timezone":"Europe/Oslo","ip":"80.>
{"clientip":"93.190.71.150","@timestamp":"2021-11-28T03:55:04.031Z","geoip":{"ip":"93.190.71.150","country_name":"Germany","country_code2":"DE","latitude":51.2993,"location":{"lon":9.491,"lat":51.2993},"longitu>
```

- In the snapshot above of the parsed output we can see the country, country_code, postal_code, latitude, longitude, timezone etc.

Therefore we have successfully parsed the nginx logs file using logstash as per our requirements.