



Install and configure EclecticIQ Platform

Installation and configuration on Ubuntu Server — For system administrators

Last generated: July 21, 2017



©2017 Eclectiq

All rights reserved. No part of this document may be reproduced in any form or by any electronic or mechanical means, including information storage and retrieval systems, without written permission from the author, except in the case of a reviewer, who may quote brief passages embodied in critical articles or in a review.

Trademarked names appear throughout this book. Rather than use a trademark symbol with every occurrence of a trademarked name, names are used in an editorial fashion, with no intention of infringement of the respective owner's trademark.

The information in this book is distributed on an "as is" basis, without warranty. Although every precaution has been taken in the preparation of this work, neither the author nor the publisher shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this book.

©2017 by Eclectiq BV. All rights reserved.
Last generated on Jul 21, 2017

Table of contents

Table of contents	2
Ubuntu installation and configuration guide	5
Scope	5
Goal	5
Audience	5
Feedback	5
Before you start	7
About EclecticIQ Platform	7
Hardware requirements	7
Single box	7
Scaling out	8
Software requirements	9
Credentials and host name	9
Operating systems	9
Third-party products	9
SELinux	11
Check SELinux status	11
Check SELinux mode	12
Set SELinux to permissive mode	12
Post-installation check	12
SELinux is not installed	13
SELinux is installed but it is not enabled	14
Whitelist URLs	16
Repositories	16
Enrichers and feeds	16
Other URLs	18
Open ports	18
Install the dependencies	20
Java Development Kit	20
Supervisor	21
Nginx	22
PostgreSQL	22
Configure PostgreSQL (1 of 2)	23
Create the database	23
Set the database authentication method	23
Restart the database service	24
Redis	24
Node.js	25
ELK stack	25
Elasticsearch	25
Autostart on boot	26
Install the Elasticsearch plugins	26
Logstash	27
Kibana	27
Neo4j	27
Postfix	28
Python packages	29
poppler-utils	29
eclecticiq user and group	29
Install via a DEB package	31
Install from the APT repository	31
Add key and repository	31
Install from repository or local DEB packages	32
Check the version	33
Check permissions	33
Copy supervisord.conf	34
Configure and bootstrap	34
Config and log files	35

Configuration, log and manifest files	35
Configuration files	35
Log files	38
Manifest files	41
Default users	43
Whitelist URLs	45
Repositories	45
Enrichers and feeds	45
Other URLs	47
Open ports	47
Configure the platform	49
Configure PostgreSQL (2 of 2)	49
Add the database to the platform settings	50
Enable the service	50
Configure Nginx	51
Enable the service	53
Configure Redis	53
Enable the service	55
Configure Elasticsearch	55
Enable the service	57
Configure Logstash	58
Enable the service	58
Configure Kibana	59
Enable the service	60
Configure Neo4j	60
Enable the service	62
Configure Postfix	62
Enable the service	64
Update the platform settings	64
Secret key and session token	66
Configure OpenTAXII	66
Configure LDAP authentication	66
Configure LDAP to work with AD	70
Example	71
Configure SAML authentication	73
Test SAML authentication	75
Bootstrap the platform	78
Load the Supervisor configuration	78
Set up the database	81
Bootstrap Elasticsearch	82
Bootstrap Neo4j	82
Prerequisites	82
Create the graph schema	83
Load the dashboard	83
Run a final check	84
Check core processes and services	84
Check search indexing and graph	85
Check availability	86
Reload Supervisor configurations	86
Access the platform	87
Backup guidelines	88
Platform configuration	88
Platform databases	90
Backup guidelines	91
Back up the PostgreSQL database	91
SQL dump	91
File system level backup	91
Continuous archiving	92
Back up the Elasticsearch database	93
Back up the Neo4j database	93

Data recovery	94
Check for failed packages	94

Ubuntu installation and configuration guide

This document guides you through the installation, setup and configuration of EclecticIQ Platform when you choose to install the product from a DEB package on a target system running Ubuntu Server 16.04.

Scope

This document guides you through the steps you need to carry out to complete the following tasks:

- Check and install third-party products and dependencies.
- Install the platform.
- Look for and identify platform configuration and log files.
- Configure the platform and the third-party components it uses.
- Bootstrap the platform before starting it for the first time.
- Launch the platform.
- Upgrade the platform to a newer release.
- Back up your data.

Goal

After completing these tasks, you'll have achieved the following goals:

- EclecticIQ Platform is installed, set up, and configured in the target system.
- The necessary dependencies and third-party products are installed and configured to work with the platform.
- The platform is ready for use.

Audience

This document targets the following audience:

- DevOps
- System administrators


Feedback

No one reads manuals, ever. We know.

Yet, we strive to give you clear, concise, and complete documentation that helps you get stuff done neatly.

We are committed to crafting good documentation, because life is too short for bad doc.

We appreciate your comments, and we'd love to hear from you: if you have questions or suggestions, drop us a line and share your thoughts with us!

 The Product Team

Before you start

Review the system requirements before proceeding to install the platform on Ubuntu from a DEB package.

About EclecticIQ Platform

EclecticIQ Platform is powered by **STIX** (<https://stixproject.github.io/>) and **TAXII** (<http://taxiiproject.github.io/about/>) open standards.

It enables ingesting, consolidating, analyzing, integrating, and collaborating on intelligence from multiple sources.

EclecticIQ Platform	Key features
Feed management	Manage multiple cyber threat intelligence feeds from any source, in many different formats.
Enrichment	Enrich existing intelligence with external data sources, and refine it with de-duplication and pattern recognition.
Sharing	Investigate and identify threats together with partners as part of an information ecosystem.
Collaboration	Analyze and create intelligence in collaboration with other teams and departments.
Insights	Generate insight thanks to a high-fidelity, normalized view into your intelligence.
Integration	Understand how cyber intelligence relates to and how it can affect your organization and your environment.

Hardware requirements

Hardware requirements for EclecticIQ Platform can vary depending on the target environment you plan to install the platform to. Therefore, the requirements outlined in this section are general guidelines that work in most cases, but they are not tailored to any specific situation.

Single box

Hardware requirement guidelines for EclecticIQ Platform and related dependencies installation on one target machine.

HW area	Minimum	Recommended	Notes
Environment	-	Physical machine/ <i>rpm</i> and <i>deb</i> installs	
CPUs	4	8	Core count includes HT

HW area	Minimum	Recommended	Notes
CPU speed	2.5 GHz	2.5 GHz or faster	
Memory	16 GB	at least 32 GB	16 GB is unsuitable for production. A production environment should feature at least 32 GB memory. Consider expanding it to 64 GB when dealing with, for example, large data corpora ingestion or data-intensive graph visualizations. Monitor system memory usage to determine if your system may need more memory to operate smoothly.
Storage	SATA, 100 IOPS	SSD, 200 IOPS	Local attached storage is preferable to SAN or NAS; platform operations are write-intensive. Recommended IOPS range: 200-500
Drives	5	10	10 drives to set up 5 sets of mirrored drives (RAID 1)
Drive sizes (GB)	10, 10, 25, 50, 200	20, 20, 50, 75, 300	Each platform database should be allocated to a dedicated drive for data storage
Drive allocation (GB)	10	20	Root (EclecticIQ Platform + Redis)
	10	20	Log data storage
	25	50	Neo4j, graph database
	50	75	Elasticsearch, searching and indexing
	200	300	PostgreSQL, main data storage
Network	2 network interfaces	2 network interfaces	1 interface for production, the other for system management
Install size	~240 GB	~240 GB	Full install, based on VM image size

Scaling out

The easiest approach to scaling out involves allocating dedicated machines to the databases. In this scenario, you install each of the following components on a separate machine:

- EclecticIQ Platform
- PostgreSQL
- Redis
- Elasticsearch
- Neo4j

To optimize read-write operations and to ensure that the storage drives are fast, set up dedicated drives per partition.

Software requirements

Credentials and host name

To correctly configure the system after installing the required dependencies and third-party products, ensure you have the following information available:

- DNS name of the host you are going to use to access the platform.
Example: `platform.host`
- SSL certificate and key for the web server.
- EclecticIQ Platform login credentials.

EclecticIQ Platform default login credentials	
user name	admin
password	EclecticIQ2015#

Operating systems

Supported operating systems:

- **Ubuntu Server 16.04 LTS (Xenial Xerus)** (<https://wiki.ubuntu.com/xenialxerus/releasenotes>)

Third-party products

Third-party software includes required dependencies for EclecticIQ Platform to operate correctly.



Make sure that the following software products are *already installed* on the target system *before* installing the platform.

During installation, the platform checks for these dependencies.

If they are missing, the installation procedure aborts.

Dependency	Version	Reference
Oracle Java JDK	1.8.0	Oracle Java download page (http://www.oracle.com/technetwork/java/javase/downloads/index.html)

Dependency	Version	Reference
PostgreSQL	9.5	PostgreSQL documentation about installing on Ubuntu (https://www.postgresql.org/download/linux/ubuntu/)
Redis	3.2.3	Redis download page (http://redis.io/download)
Nginx	1.12	Nginx download page (https://nginx.org/download/)
Neo4j	2.3.1 Community	Neo4j download page (http://neo4j.com/download/)
Elasticsearch	2.3.5	Elasticsearch installation documentation (https://www.elastic.co/guide/en/elasticsearch/reference/master/deb.html)
delete-by-query	n/a	delete-by-query installation documentation (https://www.elastic.co/guide/en/elasticsearch/plugins/2.3/plugins-delete-by-query.html)
elasticsearch-dump	2.4.2	Install globally with npm as a Node js module (https://www.npmjs.com/package/elasticsearch-dump)
Logstash	2.3.4	Logstash install instructions (https://www.elastic.co/guide/en/logstash/2.3/installing-logstash.html)
Kibana	4.5.1	Kibana install instructions (https://www.elastic.co/guide/en/kibana/4.5/setup-repositories.html)
unzip	n/a	Install with apt-get install zip unzip on Ubuntu Server (https://www.digitalocean.com/community/questions/how-to-install-zip-in-ubuntu)
Node.js	6.x or later	Node.js for Ubuntu Server (includes npm) (https://nodejs.org/en/download/package-manager/#debian-and-ubuntu-based-linux-distributions)
poppler-utils	n/a	Install with apt-get install poppler-utils on Ubuntu Server (https://packages.ubuntu.com/xenial/python/poppler-utils)
Supervisor	n/a	Install with apt-get install supervisor on Ubuntu Server (https://www.howtoinstall.co/en/ubuntu/xenial/supervisor)
Postfix	n/a	Install with apt-get install postfix on Ubuntu Server (https://help.ubuntu.com/lts/serverguide/postfix.html)

Suggested dependency installation sequence:

- Oracle Java JDK
- Supervisor
- Nginx
- PostgreSQL
- Redis
- Node.js
- Elasticsearch
- Logstash

- Kibana
- delete-by-query
- elasticsearchdump
- Neo4j
- Postfix
- unzip
- poppler-utils



- As per *elasticsearchdump* version 2.4.0, we recommended installing *elasticsearchdump* as a **global Node.js module** (<https://www.npmjs.com/package/elasticsearchdump#installing>).
- When carrying out maintenance and system administration activities on any of the required third-party platform components, first gracefully shut down the platform, and then proceed with the other tasks.



Warning: About Elasticsearch

During complex index upgrades and reindexing operations, Elasticsearch may require additional disk space to store temporary working files and temporary copies of the existing indices.

Monitor your Elasticsearch partition usage. Before it reaches 50% of the available space in the partition, extend it, so that the new partition size is at least twice as large as the sum of the existing Elasticsearch indices.

Example: if Elasticsearch currently uses 43 GB of disk space, extend the partition where Elasticsearch lives, so that it is at least 86 GB.

SELinux



EclecticIQ Platform supports **SELinux** (<http://selinuxproject.org/>).

- If you are using or plan to use SELinux in the environment where the platform is installed, you should carry out this check.
- If you are not using SELinux and are not planning to implement it in the environment where the platform is installed, you do not need to do anything and you can safely disregard this section.

Check SELinux status

If SELinux is installed, check if it is enabled or disabled. Run the following command(s):

```
$ sestatus -v
```

If SELinux is disabled, the response includes the following line:

```
SELinux status: disabled
```

Check SELinux mode

You can check also which SELinux mode is currently active. Run the following command(s):

```
$ getenforce
```

The allowed modes are enforcing, permissive, and disabled.

The active mode may not be the same as the `SELINUX` value defined in the SELinux global configuration file:

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=permissive
# SELINUXTYPE= can take one of three two values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted
```

This can happen after changing and saving **SELinux global configuration file**

(<https://selinuxproject.org/page/configurationfiles>), and before executing a system reboot for the changes to become effective.

Set SELinux to permissive mode

The recommended SELinux mode to offload complexity during installation and configuration is `permissive`.

To set SELinux to work permissively, run the following command(s):

```
$ setenforce permissive
```

Post-installation check

The platform post-install script included in the RPM package attempts to automatically set the appropriate SELinux security labels for the files that are deployed to */opt/eclecticiq*.

- If SELinux is not installed or if it is disabled, the post-install script included in the RPM install package does not attempt to configure any SELinux file security labels for the files that are deployed to */opt/eclecticiq*.
- If SELinux is installed and it is enabled, and if the platform post-install script does not set the SELinux security labels to the applicable platform files, run the following command(s):

```
$ semanage fcontext -a -t var_log_t -f d "/opt/eclecticiq"
```

- If SELinux policy-related errors occur, the command returns a response that can be similar to this example:

```
SELinux: Could not downgrade policy file /etc/selinux/targeted/policy/policy.29, searching for
an older version.
SELinux: Could not open policy file <= /etc/selinux/targeted/policy/policy.29: No such file or
directory
/sbin/load_policy: Can't load policy: No such file or directory
libsemanage.semanage_reload_policy: load_policy returned error code 2.
```

The response provides more context about the affected files and the reasons why it was not possible to set the security labels.

SELinux is not installed

If SELinux is not installed on the target system, do the following:

- After completing the platform installation, install and enable SELinux.
- To set the correct security contexts, execute the following script:

```
BASE_PATH="/opt/eclecticiq"

if [ -x "$(command -v semanage)" ]; then

    SELINUX_MODE=$(getenforce)

    if ! [ $SELINUX_MODE == "Disabled" ]; then

        semanage fcontext -a -t etc_t "$BASE_PATH/etc(/.*)?"
        semanage fcontext -a -t etc_t "$BASE_PATH/etc-extras(/.*)?"

        semanage fcontext -a -t httpd_config_t "$BASE_PATH/etc/nginx(/.*)?"
        semanage fcontext -a -t httpd_config_t "$BASE_PATH/etc-extras/nginx(/.*)?"

        # By default, newly created files and directories inherit the SELinux type
        # of the corresponding parents, so that log files have the correct type.
        # However, we do not want to relabel existing logs.
        semanage fcontext -a -t var_log_t -f d "$BASE_PATH/logs"

        restorecon -RF $BASE_PATH

        echo "SELinux security labels configured."
    else
        echo "SELinux is not enabled. Security labels won't be configured."
    fi
else
    echo "SELinux is not installed. Security labels won't be configured."
fi
```

- You may need to reboot the system for the changes to become effective.

SELinux is installed but it is not enabled

If SELinux is installed on the target system but it is not enabled, do the following:

- Enable SELinux, either by editing its configuration file, and then by rebooting the system, or by running one of the following commands:

```
# Set SELinux to permissive mode
$ setenforce 0

# Set SELinux to enforcing mode
$ setenforce 1
```

- Create the following bash script:

```
BASE_PATH="/opt/eclecticiq"

if [ -x "$(command -v semanage)" ]; then

    SELINUX_MODE=$(getenforce)

    if ! [ $SELINUX_MODE == "Disabled" ]; then

        semanage fcontext -a -t etc_t "$BASE_PATH/etc(/.*)?"
        semanage fcontext -a -t etc_t "$BASE_PATH/etc-extras(/.*)?"

        semanage fcontext -a -t httpd_config_t "$BASE_PATH/etc/nginx(/.*)?"
        semanage fcontext -a -t httpd_config_t "$BASE_PATH/etc-extras/nginx(/.*)?"

        # By default, newly created files and directories inherit the SELinux type
        # of the corresponding parents, so that log files have the correct type.
        # However, we do not want to relabel existing logs.
        semanage fcontext -a -t var_log_t -f d "$BASE_PATH/logs"

        restorecon -RF $BASE_PATH

        echo "SELinux security labels configured."
    else
        echo "SELinux is not enabled. Security labels won't be configured."
    fi
else
    echo "SELinux is not installed. Security labels won't be configured."
fi
```

- Save it, make it executable, and then run it.
- You may need to reboot the system for the changes to become effective.

Whitelist URLs

The platform needs to access external data sources to ingest intel, as well as to enrich entities and observables. You may want to whitelist these URLs, domains and addresses, so that the platform can communicate with the external intel and service providers.

Repositories

When installing or upgrading the platform and its dependencies, the system needs to access the following source repositories.

Repository URL	Belongs to	Repo type
https://downloads.eclecticiq.com/platform	EclecticIQ Platform	deb, rpm
https://dl.fedoraproject.org/pub/epel/7/x86_64/	EPEL	rpm
https://centos7.iuscommunity.org/	IUS	rpm
http://download.oracle.com/	Oracle Java JDK	deb, rpm
https://nginx.org/packages/ubuntu/pool/nginx/n/nginx/	Nginx	deb
https://nginx.org/packages/centos/7/x86_64/	Nginx	rpm
https://download.postgresql.org/pub/repos/apt/pool/9.5/	PostgreSQL	deb
https://download.postgresql.org/pub/repos/yum/9.5/redhat/	PostgreSQL	rpm
https://packages.elastic.co/	ELK stack	deb, rpm
http://debian.neo4j.org/	Neo4j	deb
http://yum.neo4j.org/	Neo4j	rpm
http://download.redis.io/releases/	Redis	deb, rpm
https://deb.nodesource.com/	Node.js	deb
https://rpm.nodesource.com/	Node.js	rpm

Enrichers and feeds

Feeds and enrichers access data sources through these URLs. Whitelist the domains and allow traffic to and from them.

Domain	Belongs to	Type
<a href="http://<elasticsearch_url>:9200/<schema_resource>">http://<elasticsearch_url>:9200/<schema_resource>	Elasticsearch sightings	enricher/API

Domain	Belongs to	Type
https://cybercrime-portal.fox-it.com/	Fox-IT InTELL Portal	enricher/API, incoming feed/API
https://api.intel471.com/v1/	Intel 471	enricher/API, incoming feed/API
http://api.openresolve.com/{}/{}	OpenDNS OpenResolve	enricher/API
http://10.0.1.60:8000/ (example)	PyDat	enricher/API
https://stat.ripe.net/data/geoloc/data.json?resource={IP_address}	RIPEstat GeolIP	enricher/API
https://stat.ripe.net/data/whois/data.json?resource={IP_address}	RIPEstat Whois	enricher/API
https://panacea.threatgrid.com/api/v2/	Cisco Threat Grid	enricher/API, incoming feed/API
https://www.virustotal.com/vtapi/v2/{}	VirusTotal	enricher/API
https://endlesstunnel.info/v3	Flashpoint AggregINT	enricher/API
https://endlesstunnel.info/v3	Flashpoint Blueprint	enricher/API
https://endlesstunnel.info/v3	Flashpoint Thresher	enricher/API
https://api.passivetotal.org/v2	PassiveTotal Whois	enricher/API
https://api.passivetotal.org/v2	PassiveTotal Passive DNS	enricher/API
https://api.passivetotal.org/v2	PassiveTotal IP/Domain	enricher/API
https://api.passivetotal.org/v2	PassiveTotal Malware	enricher/API
http://10.0.1.22:8089/ (example)	Splunk sightings	enricher/API
http://api.domaintools.com/v1/{}/host-domains	DomainTools Hosted Domains	enricher
http://api.domaintools.com/v1/reputation	DomainTools Reputation	enricher
https://api.domaintools.com/v1/{}/host-domains	DomainTools Suspicious Domains	enricher
https://api.isightpartners.com/search/{}	FireEye iSIGHT	enricher
https://app.recordedfuture.com/live/sc/entity/{}	Recorded Future	enricher
https://unshorten.me/s/{}	Unshorten-URL	enricher
https://api.dnsdb.info/{}	Farsight DNSDB	enricher
https://www.threatcrowd.org/{}	ThreatCrowd	enricher
https://censys.io/api/v1/search/ipv4	Censys	enricher
http://api.domaintools.com/v1/{}/name-server-domains/	DomainTools Malicious Server Domains	enricher

Domain	Belongs to	Type
http://api.domaintools.com/v1/{}/whois/parsed	DomainTools Retrieve Parsed Whois Observables	enricher
https://intelapi.crowdstrike.com/indicator/v1/search/{}	CrowdStrike Falcon Intelligence Indicator	enricher

Other URLs

Domain	Belongs to	Type
http://<variable_subdomain>.cyberfeed.net:<port_number>	AnubisNetworks	incoming feed
https://api.threatrecon.co/	Threat Recon	incoming feed
http://hailataxii.com	Hail a TAXII	open source cyber threat intelligence source
https://test.taxiistand.com/	TAXII Stand	public OpenTAXII test server

Open ports

The platform components communicate with the platform and with each other through these ports. Make sure they are open within the platform network.

Port	Belongs to
9200	elasticsearch
5601	kibana
7474	neo4j
4008	neo4j-batching
8008	platform-api
5432	postgresql-9.5
6379	redis
6755	logstash
80; 443	nginx
25; 587	postfix
9001	supervisor

Install the dependencies

Install all required dependencies and third-party components before installing the platform.

To successfully execute several commands in the command line or in the terminal, you may need root-level access rights.

To obtain admin rights, run the following command(s):

```
$ sudo su -
```

Alternatively:

- Grant admin rights to a specific user, who can then log in with their password to perform admin tasks:

```
$ su - {user_name}
```

Or:

- Prefix `sudo` to the command you want to run:

```
$ sudo {command}
```

Java Development Kit

- To install **Oracle Java SE Development Kit 8** (<http://www.oracle.com/technetwork/java/javase/downloads/>), add the repository to the list of configured sources by running the following command(s):

```
$ add-apt-repository ppa:webupd8team/java
```

- Download and update package information for all configured sources:

```
$ apt-get update
```

- Install Oracle JDK 8 by running the following command(s):

```
$ apt-get install oracle-java8-installer
```

- Verify the JDK version installed on the target system:

```
$ java -version
```

You may need to verify that the `JAVA_HOME` environment variable in `/etc/environment` is correctly set. If you need to set it manually, do the following:

- Identify the location of the JDK installation:

```
$ update-alternatives --config java
```

- Copy the path from the JDK 8 installation.
- In a text editor, open `/etc/environment`:

```
$ nano /etc/environment
```

- Add the following line to the open file (replace the dummy path in the example with the actual path to the JDK install on your system):

```
JAVA_HOME="/path/to/jdk-install"
```

- Save and close the file, and then reload it by running the following command(s):

```
$ source /etc/environment
```

If necessary, you can add `source /etc/environment` to `~/.bashrc` to load all environment settings at startup.

- Check the `JAVA_HOME` environment variable by running the following command(s):

```
$ echo $JAVA_HOME
```

Supervisor

Supervisor (<http://supervisord.org/>) is a process manager the platform uses to manage processes and services.

- To install **Supervisor** (<https://www.digitalocean.com/community/tutorials/how-to-install-and-manage-supervisor-on-ubuntu-and-debian-vps>) run the following command(s):

```
$ apt-get install supervisor
```

- After completing the installation, manually restart Supervisor by first stopping it, and then by starting it. In this way, it can pick up any changed configurations before starting all managed applications:

```
$ systemctl stop supervisor
```

```
$ systemctl start supervisor
```

Nginx

Nginx (<https://www.digitalocean.com/community/tutorials/how-to-install-nginx-on-ubuntu-16-04>) is the web server the platform uses.

- Add the Nginx repository to the list of configured sources by running the following command(s):

```
$ sudo -s
$ nginx=stable
$ add-apt-repository ppa:nginx/$nginx
```

- Download and update package information for all configured sources:

```
$ apt-get update
```

- **Install** (<https://www.nginx.com/resources/wiki/start/topics/tutorials/install/#official-debian-ubuntu-packages>) **Nginx 1.12**, the current stable version, by running the following command(s):

```
$ apt-get install nginx
```

- Verify that Nginx is correctly installed:

```
$ type nginx
```

PostgreSQL

- Download and update package information for all configured sources:

```
$ apt-get update
```

- **Install** (<https://www.tqhosting.com/kb/428/how-to-install-postgresql-95-on-ubuntu-1604-lts-xenial-xerus.html>) **PostgreSQL 9.5** by running the following command(s):

```
$ apt-get install postgresql-9.5 -y
```

- Install these additional PostgreSQL packages by running the following command(s):

```
$ apt-get install postgresql-contrib-9.5 postgresql-server-dev-9.5
```

- Enable the PostgreSQL service to start at system bootup:

```
$ systemctl enable postgresql
```

- To start PostgreSQL, run the following command(s):

```
$ systemctl start postgresql
```

- To check if PostgreSQL is running, run the following command(s):

```
$ systemctl status postgresql
```

Configure PostgreSQL (1 of 2)



Warning:

You should configure PostgreSQL in 2 steps:

- Create the main data storage *before* installing the platform.
- Add the database to the platform settings *after* installing the platform.

Create the database

- Connect to PostgreSQL with root privileges, and access it with the `postgres` user name:

```
$ su - postgres  
$ psql
```

- In PostgreSQL do the following:
 - **Create** (<http://www.postgresql.org/docs/current/static/tutorial-createdb.html>) a new database (in the example: *platform*).
 - Create a new user, and assign them a password (in the example: *test/test*, respectively).
 - Grant the new user full privileges on the newly created database.

```
CREATE DATABASE platform;  
CREATE USER test WITH PASSWORD 'test';  
GRANT ALL PRIVILEGES ON DATABASE platform to test;
```

Set the database authentication method

- Open the `pg_hba.conf` configuration file in a text editor:

```
$ nano /etc/postgresql/9.5/main/pg_hba.conf
```


- Set the authentication method to `trust` for the `local` type, and to `md5` for the `host` type to request an MD5-encrypted password:

#	TYPE	DATABASE	USER	ADDRESS	METHOD
	<code>local</code>	<code>all</code>	<code>test</code>		<code>trust</code>
	<code>host</code>	<code>all</code>	<code>all</code>	<code>same-net</code>	<code>md5</code>

Restart the database service

- Restart the PostgreSQL service:

```
$ systemctl restart postgresql
```

- Verify that user and database are successfully created:

```
$ psql -U test -d platform -W
```

Redis

Redis acts as a message broker for Celery-managed tasks and ingestion queues.

To install **Redis** (<http://redis.io/download>) do the following:

- Add the Redis repository to the list of configured sources by running the following command(s):

```
$ add-apt-repository ppa:chris-lea/redis-server
```

- Download and update package information for all configured sources:

```
$ apt-get update
```

- Install Redis:

```
$ apt-get install redis-server
```

- Verify that Redis is correctly installed:

```
$ type redis-server
$ type redis-cli
```

- If you want to check the version of the Redis server, run the following command(s):

```
$ redis-server -v
```

- If you want to check the version of the Redis client, run the following command(s):

```
$ redis-cli -v
```

- You can also invoke the **client** (<https://redis.io/commands/info>) to check the version of the server:

```
$ redis-cli  
{redis_ip:redis_port}> INFO
```

Node.js

To install **Node.js 6.x or later** (<https://nodejs.org/en/download/package-manager/#debian-and-ubuntu-based-linux-distributions>) do the following:

- Download Node.js 6.x or later:

```
$ curl -sL https://deb.nodesource.com/setup_6.x | sudo -E bash -
```

- Install Node.js 6.x or later:

```
$ apt-get install -y nodejs
```

- Verify that Node.js is correctly installed:

```
$ type node
```

ELK stack

Elasticsearch

- Download and install the public GPG signing key:

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

- Add the Elasticsearch repository to the list of configured sources by running the following command(s):

```
$ echo "deb http://packages.elastic.co/elasticsearch/2.x/debian stable main" | sudo tee -a /etc/apt/sources.list.d/elasticsearch-2.x.list
```

- Download and update package information for all configured sources:

```
$ apt-get update
```

- Install **Elasticsearch 2.3.5** (<https://www.elastic.co/guide/en/elasticsearch/reference/master/deb.html>):

```
$ apt-get install -y elasticsearch=2.3.5
```

Autostart on boot

- Enable Elasticsearch and set it up to automatically start on system boot:

```
$ systemctl daemon-reload  
$ systemctl enable elasticsearch
```

- Start the Elasticsearch service:

```
$ systemctl start elasticsearch
```

- Verify that the Elasticsearch service is up and running:

```
$ systemctl status elasticsearch
```

Install the Elasticsearch plugins

- Install *delete-by-query* by running the following command(s):

```
$ /usr/share/elasticsearch/bin/plugin install delete-by-query
```

In case the standard installation fails because of a Java CA certificate error, retry the operation by running the following command(s):

```
$ /var/lib/dpkg/info/ca-certificates-java.postinst configure  
  
$ /usr/share/elasticsearch/bin/plugin install delete-by-query -vv
```

- Install *elasticsearch-dump* 2.4.2:

```
$ npm install elasticsearch-dump@2.4.2 -g
```

Logstash

- Add the Logstash repository to the list of configured sources by running the following command(s):

```
$ echo "deb https://packages.elastic.co/logstash/2.3/debian stable main" | sudo tee -a /etc/apt/sources.list
```

- Download and update package information for all configured sources:

```
$ apt-get update
```

- Install **Logstash 2.3.4** (<https://www.elastic.co/guide/en/logstash/2.3/installing-logstash.html>):

```
$ apt-get install -y logstash=1:2.3.4-1
```

Kibana

- Add the Kibana repository to the list of configured sources by running the following command(s):

```
$ echo "deb http://packages.elastic.co/kibana/4.5/debian stable main" | sudo tee -a /etc/apt/sources.list
```

- Download and update package information for all configured sources:

```
$ apt-get update
```

- Install **Kibana 4.5.1** (<https://www.elastic.co/guide/en/kibana/4.5/setup-repositories.html>):

```
$ apt-get install -y kibana=4.5.1
```

- Follow the steps outlined in the **official documentation** (<https://www.elastic.co/downloads/kibana>) and review the product **Readme file** (<https://github.com/elastic/kibana/blob/4.5/readme.md>).

Neo4j

- Download and install the public PGP signing key:

```
$ wget -O - https://debian.neo4j.org/neotechnology.gpg.key | sudo apt-key add -
```

- Add the Neo4j repository to the list of configured sources by running the following command(s):

```
$ echo 'deb https://debian.neo4j.org/repo stable/' | sudo tee /etc/apt/sources.list.d/neo4j.list
```

- Download and update package information for all configured sources:

```
$ apt-get update
```

- **Install** (<http://codegist.net/search?q=install+neo4j+on+ubuntu>) **Neo4j 2.3.1 Community** (<https://debian.neo4j.org/>):

```
$ apt-get install -y neo4j=2.3.1
```

- In the system home directory, open the `.bash_profile` file in a text editor:

```
$ nano ~/.bash_profile
```

- Add the following lines to the `.bash_profile` file:

```
NEO4J_HOME=/usr/share/neo4j  
PATH=$PATH:$HOME/bin:$NEO4J_HOME
```

After installing the platform, make sure these environment variables are accessible to the `neo4j` user, `neo4j` group profile.

Postfix

- Download and update package information for all configured sources:

```
$ apt-get update
```

- **Install** (<https://help.ubuntu.com/lts/serverguide/postfix.html>) **Postfix** (<http://www.postfix.org/>):

```
$ DEBIAN_PRIORITY=low apt-get install postfix
```

For a thorough tutorial and walkthrough to install and configure Postfix on Ubuntu Server 16.04, see **How to install and configure Postfix on Ubuntu 16.04** (<https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-postfix-on-ubuntu-16-04>).

Python packages

- Check if Python is installed on the target system:

```
# Check for Python 2.x:
$ python --version

# Example response:
Python 2.7.12+

# Check for Python 3.x:
$ python3 --version

# Example response:
Python 3.5.2+
```

- If you need to install Python on the system, run the following command(s):

```
$ apt-get install python python3 python-dev python3-dev
```

- Do install the following Python packages, as they are *platform dependencies*:

```
$ apt-get install libxml2-dev python-libxslt1 libyaml-dev python-yaml python-psycpg2
```

poppler-utils

The platform requires **poppler-utils** (<https://poppler.freedesktop.org/>) to process PDF ([https://en.wikipedia.org/wiki/poppler_\(software\)#poppler-utils](https://en.wikipedia.org/wiki/poppler_(software)#poppler-utils)) files.

- Install **poppler-utils** (<https://packages.ubuntu.com/xenial/python/poppler-utils>) by running the following command(s):

```
$ apt-get install poppler-utils
```

eclecticiq user and group

- Before installing the platform, create an `eclecticiq` user and group, and assign the `eclecticiq` user to the group with the same name:

```
$ useradd -c "EclecticIQ Platform" -s /bin/bash -m -U eclectic
```


Install via a DEB package

After checking the necessary requirements and dependencies the platform needs to work, you can proceed to install the platform.

To successfully execute several commands in the command line or in the terminal, you may need root-level access rights.

To obtain admin rights, run the following command(s):

```
$ sudo su -
```

Alternatively:

- Grant admin rights to a specific user, who can then log in with their password to perform admin tasks:

```
$ su - {user_name}
```

Or:

- Prefix `sudo` to the command you want to run:

```
$ sudo {command}
```

Install from the APT repository



Before beginning the installation procedure, make sure you have the necessary login credentials to access the provided resource download locations on Bintray.

Add key and repository

You can add a PGP key *either* from a local file, *or* by downloading it.

Add the key from a saved file

To install EclecticIQ Platform do the following:

- Download and install the public PGP signing key:
 - If you receive the key as a `.key` file, download it and save it locally.
Example: `repomd.xml.key`
 - Add the saved key by running the following command(s):


```
$ apt-key add /path/to/key/key_filename.key

# Successful response:
OK
```

Download the key

- If you retrieve the key through a link, download it and add it by running the following command(s):

```
$ wget -qO - https://{user_name}:{password}@downloads.eclecticiq.com/eiq-prerequisites-rpm/2.0/repodata/repomd.xml.key | sudo apt-key add -
```

Add the repository

- Add the EclecticIQ Platform repository to the list of configured sources by running the following command(s):

```
$ echo "deb https://{user_name}:{password}@downloads.eclecticiq.com/eiq-platform-deb xenial 1.14.4" | sudo tee -a /etc/apt/sources.list.d/eclecticiq-platform.list
```

- Download and update package information for all configured sources:

```
$ apt-get update
```

Install from repository or local DEB packages

You can install EclecticIQ Platform *either* from a repository, when available, *or* from locally saved *.deb* packages.

Install from repository

- Install EclecticIQ Platform 1.14.4 by running the following command(s):

```
$ apt-get install eclecticiq-platform
```

Install from local .deb packages

- Download and save locally the necessary *.deb* packages to install all EclecticIQ Platform components.
Example: */tmp/eclecticiq-platform-deb*
- Go to the directory where the *.deb* packages are stored

```
# Example save-to directory
$ cd /tmp/eclecticiq-platform-deb
```

- In the directory run the following command(s):

```
# Example save-to directory
$ apt install /tmp/eclecticiq-platform-deb/*.deb
```

Alternatively, you can install local `.deb` packages with **gdebi** (<https://packages.ubuntu.com/xenial/gdebi>), which features both a GUI and a command line.

Check the version

Before starting the installation, a script carries out a version check to detect any previously installed platform versions.

It is possible to upgrade the platform only sequentially, it is not possible to skip versions.

For example, if you want to upgrade from release 1.10.0 to 1.14.1, you need to upgrade step by step by installing all intermediate releases until 1.14.1 included.

If the currently installed version details are the same as the corresponding information in the upgrade version, or if the former does not immediately precedes the latter, the upgrade procedure is aborted, and an error message is displayed:

```
You currently have version ${INSTALLED_VERSION}, but you need to have ${PREV_VERSION} installed
in order to upgrade ${PACKAGE_NAME}.

exit 99
```

<code>\${INSTALLED_VERSION}</code>	The currently installed version on your system. The upgrade fails because this version is too old.
<code>\${PREV_VERSION}</code>	Before starting the upgrade procedure again, install this platform version on your system. The upgrade procedure works only when the old and the new versions are contiguous in sequence.
<code>exit 99</code>	The exit code accompanying the error message.

Check permissions

Make sure the target directory to save application data to exists before setting it in the configuration file, and that it is accessible to read and write data.

If necessary, do the following:

- Create the target directory:

```
$ mkdir /<path_to_dir>/<target_dir>
```

- Make sure the correct user owns the target directory to access it in read and write modes.

`eclecticiq:eclecticiq` should own Supervisor configuration and log file locations:

```
$ chown -R eclecticiq:eclecticiq /var/run/supervisor/
```

```
$ chown -R eclecticiq:eclecticiq /etc/supervisord.d/
```

```
$ chown -R eclecticiq:eclecticiq /var/log/eclecticiq
```



If you reboot the system after applying these changes, ownership of the `/var/run/supervisor/` directory may be reset to `root:root`.

In this case, reassign ownership back to `eclecticiq:eclecticiq`:

```
$ chown -R eclecticiq:eclecticiq /var/run/supervisor/
```

Copy supervisord.conf

- Copy the Supervisor configuration file provided with the platform, so that it overwrites the default Supervisor *supervisord.conf* file:

```
$ cp /opt/eclecticiq/etc-extras/supervisord/supervisord.conf /etc/supervisor/supervisord.conf
```

Configure and bootstrap

After a successful platform installation, the following message is displayed to inform you that you need to carry out a few post-installation tasks to complete the upgrade procedure:

You are almost there, just a few more steps before you complete the EclecticIQ Platform upgrade procedure.

Refer to the upgrade section **in** the EclecticIQ Platform installation and configuration guide to learn what you should **do** next.



Follow the next sections in this document to proceed to the configuration and bootstrapping steps.

Config and log files

An overview of all platform configuration, log, and manifest files for system administrators.

Configuration, log and manifest files

EclectiQ Platform uses several configuration files to store platform settings you can edit and fine-tune to adapt the behavior of the platform to your system.

Log files record platform events; they hold a history of the platform activities that can provide meaningful context, for example when investigating the possible root causes of a problem.

Manifest files contain metadata that help identify the product like the source/origin of the package containing the platform and its components, release reference number, and version information.

This section describes where the platform configuration, log, and manifest files are stored, and what kind of information each file holds.

Configuration files

To get a list with the platform configuration files, run the following command(s):

```
# Returns only core config files
$ find /opt/eclecticiq/etc/eclecticiq/ -type f

# Returns platform and third-party components config files
$ find /opt/eclecticiq/etc/ -type f

# Returns third-party config files to use for reference as boilerplates
$ find /opt/eclecticiq/etc-extras/ -type f
```

The response returns a list with the following files:

```

# Core platform settings
/opt/eclecticiq/etc/eclecticiq/opentaxii.yml
/opt/eclecticiq/etc/eclecticiq/platform_settings.py
/opt/eclecticiq/etc/eclecticiq/proxy_url

# Dashboard settings
/opt/eclecticiq/etc/kibana-dashboard.json

# Logstash log aggregation settings
/opt/eclecticiq/etc/logstash/conf.d/filters.conf
/opt/eclecticiq/etc/logstash/conf.d/input.conf
/opt/eclecticiq/etc/logstash/conf.d/neo4j-batching.conf
/opt/eclecticiq/etc/logstash/conf.d/opentaxii.conf
/opt/eclecticiq/etc/logstash/conf.d/output.conf
/opt/eclecticiq/etc/logstash/conf.d/platform-api.conf
/opt/eclecticiq/etc/logstash/conf.d/platform-ui.conf

# Logstash heap size
/opt/eclecticiq/etc/default/logstash # Ubuntu
/opt/eclecticiq/etc/sysconfig/logstash # CentOS, RHEL

# Web server, proxy and port settings
/opt/eclecticiq/etc/nginx/conf.d/platform.conf

# Platform components ini file settings: ingestion, search, graph, taxii, api, tasks
/opt/eclecticiq/etc/supervisord.d/graph-ingestion.ini
/opt/eclecticiq/etc/supervisord.d/intel-ingestion.ini
/opt/eclecticiq/etc/supervisord.d/neo4j-batching.ini
/opt/eclecticiq/etc/supervisord.d/opentaxii.ini
/opt/eclecticiq/etc/supervisord.d/platform-api.ini
/opt/eclecticiq/etc/supervisord.d/search-ingestion.ini
/opt/eclecticiq/etc/supervisord.d/task-workers.ini

# Kibana config file
/opt/eclecticiq/etc-extras/kibana.yml

# Redis message broker config file
/opt/eclecticiq/etc-extras/redis.conf

# Elasticsearch config files
/opt/eclecticiq/etc-extras/elasticsearch/elasticsearch.yml
/opt/eclecticiq/etc-extras/elasticsearch/logging.yml

# Neo4j config files
/opt/eclecticiq/etc-extras/neo4j/neo4j.properties
/opt/eclecticiq/etc-extras/neo4j/neo4j-server.properties
/opt/eclecticiq/etc-extras/neo4j/neo4j-wrapper.conf

# Nginx web server config file
/opt/eclecticiq/etc-extras/nginx/nginx.conf

# Postfix email server config file
/opt/eclecticiq/etc-extras/postfix/main.cf

# Kibana ini file
/opt/eclecticiq/etc-extras/supervisord/kibana.ini

```

The following table gives an overview of what information each configuration file holds.

File name and location	Description
------------------------	-------------

File name and location	Description
/opt/eclecticiq/etc/eclecticiq/platform_settings.py	Contains core platform settings like security key value, auth URLs pointing to external components Celery-managed tas
/opt/eclecticiq/etc/eclecticiq/opentaxii.yml	Contains OpenTAXII (https://opentaxii.readthedocs URL and port for the service, as well as the designated inb
/opt/eclecticiq/etc/eclecticiq/proxy_url	Contains the IP addresses and host names that should by comma-separated. The no-proxy list needs to always inclu 127.0.0.1,localhost.
/opt/eclecticiq/etc/kibana-dashboard.json	Contains the configuration defining the Kibana dashboard I web-based GUI.
/opt/eclecticiq/etc/logstash/conf.d/platform-api.conf	Defines the locations of the log files storing log data relatec API, intel ingestion, and tasks.
/opt/eclecticiq/etc/logstash/conf.d/platform-ui.conf	Defines the locations of the log files storing log data relatec Nginx is the web server; if it is not working normally or if it i may be unavailable.
/opt/eclecticiq/etc/logstash/conf.d/input.conf	Elasticsearch Curator input configuration file. Elasticsearc (https://www.elastic.co/guide/en/elasticsearch/c manages Elasticsearch indices.
/opt/eclecticiq/etc/logstash/conf.d/output.conf	Defines the Elasticsearch cluster and the data transfer prot to Elasticsearch.
/opt/eclecticiq/etc/logstash/conf.d/filters.conf	Defines filters as needed to select specific indices or data t
/opt/eclecticiq/etc/logstash/conf.d/neo4j-batching.conf	Configuration file of the neo4j-batching process.
/opt/eclecticiq/etc/logstash/conf.d/opentaxii.conf	Defines the locations of the log files storing log data relatec Nginx is the web server OpenTAXII relies on.
/opt/eclecticiq/etc/nginx/conf.d/platform.conf	Defines the platform configuration the Nginx web server ne key, ports, endpoints to make available services like the T/ documentation, and so on.
/opt/eclecticiq/etc/sysconfig/logstash	INI configuration file defining the Logstash heap size. Defa GB). Location on CentOS and RHEL OSs: /opt/eclecticiq/e /opt/eclecticiq/etc/default/logstash.
/opt/eclecticiq/etc/supervisord.d/platform-api.ini	Supervisor INI configuration file to start the platform core s
/opt/eclecticiq/etc/supervisord.d/graph-ingestion.ini	Supervisor INI configuration file to start the graph ingestion
/opt/eclecticiq/etc/supervisord.d/intel-ingestion.ini	Supervisor INI configuration file to start the intel ingestion c
/opt/eclecticiq/etc/supervisord.d/search-ingestion.ini	Supervisor INI configuration file to start the Elasticsearch ir
/opt/eclecticiq/etc/supervisord.d/neo4j-batching.ini	Initializes the neo4j-batching process. By default, the pro

File name and location	Description
/opt/eclecticiq/etc/supervisord.d/opentaxii.ini	Supervisor INI configuration file to start OpenTAXII as the incoming and outgoing feeds.
/opt/eclecticiq/etc/supervisord.d/task-workers.ini	Supervisor INI configuration file to start the Celery-managed integrations, enrichers, utilities, and beat.
/opt/eclecticiq/etc-extras/kibana.yml	Kibana configuration file. Check it and edit or update it, if not a newer version.
/opt/eclecticiq/etc-extras/redis.conf	Defines the configuration for the Redis message broker. It includes dataset snapshot autosave time intervals, and so on.
/opt/eclecticiq/etc-extras/elasticsearch/elasticsearch.yml	Elasticsearch configuration file. Check it and edit or update it to provide enough memory to the heap size (https://www.elastic.co/guide/en/elasticsearch/guide/current/_heap_memory_size.html) based on your system configuration and the size of the Elasticsearch recommended value is 2 GB : ES_HEAP_SIZE=2g
/opt/eclecticiq/etc-extras/elasticsearch/logging.yml	Configures logging and sets the logging level (info, warning, error, debug).
/opt/eclecticiq/etc-extras/neo4j/neo4j.properties	Defines the maximum size for page cache and log file
/opt/eclecticiq/etc-extras/neo4j/neo4j-server.properties	Defines the configuration options of the Neo4j graph database communication port, and so on.
/opt/eclecticiq/etc-extras/neo4j/neo4j-wrapper.conf	Defines the Java settings related to Neo4j like minimum and maximum heap size for graph database. Default min. and max. values: 4096 MB.
/opt/eclecticiq/etc-extras/nginx/nginx.conf	Defines the configuration options for the Nginx web server.
/opt/eclecticiq/etc-extras/supervisord/kibana.ini	Initializes the kibana process. By default, the process is configured to start automatically.
/opt/eclecticiq/etc-extras/neo4j-console.ini	Initializes the neo4j-console process, that is, the graph database console to start automatically.
/etc/supervisord.conf	Contains the Supervisor configuration (http://supervisord.org/configuration.html#inetd-compatible) to enable supervisord and supervisorctl. This file should include enable (http://supervisord.org/configuration.html#inetd-compatible) enabling the supervisord server to reply to system health checks.
/var/lib/pgsql/9.5/data/ (CentOS, RHEL); /etc/postgresql/9.5/main/ (Ubuntu) postgresql.conf	PostgreSQL configuration file (https://www.postgresql.org/docs/9.5/static/postgresql.conf.html).
/opt/eclecticiq/etc-extras/postfix/main.cf	Postfix configuration file (http://www.postfix.org/docs/postfix.3.3.0.html) configuring email addresses through the GUI, you need to define the host and SMTP in this file.

Log files

To get a list with the log files created by the platform and its components, run the following command(s):

```
$ find /var/log -type f
```

The response returns a list with the following files:

```
# Platform core services and components logs:
# API, ingestion, graph, OpenTAXII, Celery-managed task workers
/var/log/eclecticiq/graph-ingestion.log
/var/log/eclecticiq/intel-ingestion.log
/var/log/eclecticiq/kibana.log
/var/log/eclecticiq/neo4j-batching.log
/var/log/eclecticiq/neo4j.log
/var/log/eclecticiq/opentaxii.log
/var/log/eclecticiq/platform-api.log
/var/log/eclecticiq/search-ingestion.log
/var/log/eclecticiq/task-beat.log
/var/log/eclecticiq/task-worker-enrichers.log
/var/log/eclecticiq/task-worker-integrations.log
/var/log/eclecticiq/task-worker-providers.log
/var/log/eclecticiq/task-worker-reindexing.log
/var/log/eclecticiq/task-worker-utilities.log

# Logstash log data aggregation logs
/var/log/logstash/logstash.err
/var/log/logstash/logstash.log
/var/log/logstash/logstash.stdout

# Elasticsearch search indexing logs
/var/log/elasticsearch/intel.log
/var/log/elasticsearch/intel_deprecation.log
/var/log/elasticsearch/intel.log.YYY-MM-DD.log
/var/log/elasticsearch/intel_index_indexing_slowlog.log
/var/log/elasticsearch/intel_index_search_slowlog.log

# Nginx web server logs
/var/log/nginx/access.log
/var/log/nginx/eclecticiq-platform-ui-nginx-access.log
/var/log/nginx/eclecticiq-platform-ui-nginx-errors.log
/var/log/nginx/error.log

# PostgreSQL intel database log
/var/log/postgresql/postgresql-YYYY-MM-DD.log

# Redis message broker log
/var/log/redis/redis.log
```

The following table gives an overview of what information each log file holds.

File name and location	Description
/var/log/eclecticiq/platform-api.log	Platform log file. It logs core platform information.
/var/log/eclecticiq/graph-ingestion.log	Neo4j graph database log file. It logs information on graph database migrations, indices, and graph ingestion queue.
/var/log/eclecticiq/intel-ingestion.log	Intel ingestion log file. It logs information on ingestion events, as well as ingested batches and blobs.

File name and location	Description
/var/log/eclecticiq/search-ingestion.log	Search indexing log file. It logs information on Elasticsearch data indexing.
/var/log/eclecticiq/kibana.log	It logs information on Kibana dashboard like connection and availability.
/var/log/eclecticiq/neo4j.log	Neo4j graph database log file. It logs information on Neo4j server and database operation.
/var/log/eclecticiq/opentaxii.log	It logs OpenTAXII server log information.
/var/log/eclecticiq/task-beat.log	It logs heartbeat timestamp information. The heartbeat interval is 30 seconds.
/var/log/eclecticiq/task-worker-reindexing.log	It lists synced enricher tasks, intel providers, feed transport types, and platform utility tasks. Indexing and syncing go through Redis and Celery.
/var/log/eclecticiq/task-worker-enrichers.log	It lists enricher tasks, intel providers, feed transport types, and platform utility tasks running in the background.
/var/log/eclecticiq/task-worker-integrations.log	It lists integrations like enricher tasks, intel providers, and platform utility tasks running in the background.
/var/log/eclecticiq/task-worker-providers.log	It lists integrations like enricher tasks, intel providers, and platform utility tasks running in the background.
/var/log/eclecticiq/task-worker-utilities.log	It lists integrations like enricher tasks, intel providers, and platform utility tasks running in the background.
/var/log/logstash/logstash.err	It logs Logstash errors and error messages.
/var/log/logstash/logstash.log	It logs Logstash events.
/var/log/logstash/logstash.stdout	Standard output format for Logstash log information.
/var/log/elasticsearch/intel.log	Elasticsearch log file. It logs Elasticsearch events like initialization, startup, designated Elasticsearch cluster, and so on.
/var/log/elasticsearch/intel.log.YYYY-MM-DD.log	Elasticsearch log file for a specific date. YYYY-MM-DD (year, month, day) in the file name is replaced by the date the log information refers to. It logs Elasticsearch events like initialization, startup, designated Elasticsearch cluster, and so on.
/var/log/elasticsearch/intel_index_indexing_slowlog.log	Elasticsearch log file. It logs Elasticsearch indexing information.
/var/log/elasticsearch/intel_index_search_slowlog.log	Elasticsearch log file. It logs Elasticsearch search index information.

File name and location	Description
/var/log/postgresql/postgresql-YYYY-MM-DD.log	PostgreSQL log file. It logs PostgreSQL database intel ingestion information.
/var/log/redis/redis.log	Redis log file. It logs message broker event information about memory usage during copy-write operations and data saving to the database.
/var/log/nginx/access.log	Nginx log file. It logs web server access information.
/var/log/nginx/eclecticiq-platform-ui-nginx-access.log	Nginx log file. It logs web server access information related to the platform web-based GUI.
/var/log/nginx/eclecticiq-platform-ui-nginx-errors.log	Nginx log file. It logs web server error information related to the platform web-based GUI.
/var/log/nginx/error.log	Nginx log file. It logs web server error information.
/var/log/--	This is the root directory where all log files generated by the platform and its (third-party) components are stored.

Manifest files

To get a list with the manifest files created by the platform and its components during the installation operation, run the following command(s):

```
$ find /opt/eclecticiq/manifests -type f
```

The response returns a list with the following files:

```
# Platform manifest files:
/opt/eclecticiq/manifests/platform-api.mf
/opt/eclecticiq/manifests/platform-database.mf
/opt/eclecticiq/manifests/platform-ui.mf
```

The following table gives an overview of the metadata each manifest file holds.

File name and location	Description
/opt/eclecticiq/manifests/platform-api.mf	Manifest file with platform API metadata like package source/origin, release reference number, and version information.
/opt/eclecticiq/manifests/platform-database.mf	Manifest file with platform database release metadata like package source/origin.
/opt/eclecticiq/manifests/platform-ui.mf	Manifest file with platform GUI metadata like package source/origin, release reference number, and version information.

Default users

An overview of the default user profiles that are created during a clean platform installation.

The installation procedure creates several default user profiles at platform level, as well as at host system level, to access and manage third-party components and processes.

These users receive a standard set of user rights and permissions to allow them to carry out their tasks. They interact only with the component(s) they manage and control. In other words, these users and groups are organized in separate compartments, where each user is responsible for one or more specific, and closely related, tasks.

User	Group	Sudo	Component	Description	Home dir
eclecticiq	eclecticiq	✖	Celery workers and task runners, graph ingestion, intel ingestion, search ingestion	Platform user responsible for operational tasks like accessing Celery tasks, writing data to the graph ingestion storage location, and accessing the TAXII service	/opt/eclecticiq
elasticsearch	elasticsearch	✖	Elasticsearch search and indexing database	Search and indexing database user	/home/elasticsearch
logstash	logstash	✖	Logstash log aggregator	Log aggregator user	/opt/logstash
neo4j	neo4j	✖	Neo4j graph database	Graph database user	/usr/share/neo4j
nginx (CentOS/RHEL); www-data (Ubuntu)	nginx (CentOS/RHEL); www-data (Ubuntu)	✖	Nginx web server	Web server user	/var/cache/nginx
postgres	postgres	✖	PostgreSQL database	Database user, can access the default platform-api database	/var/lib/pgsql
redis	redis	✖	Redis server, message broker and queue manager	Redis database and message broker user	/var/lib/redis

When performing system tasks such as installation, upgrade, or maintenance, you may need to access files and directories with a specific user access profile:

```
# run this command to login as root with current user/pw
$ sudo su -

# run this command to login as root with eclecticiq user/pw
$ su - eclecticiq

# run this command to login as root with elasticsearch user/pw
$ su - elasticsearch

# run this command to login as root with logstash user/pw
$ su - logstash

# run this command to login as root with neo4j user/pw
$ su - neo4j

# run this command to login as root with nginx user/pw
$ su - nginx

# run this command to login as root with postgres user/pw
$ su - postgres

# run this command to login as root with redis user/pw
$ su - redis
```

To view platform user profiles, go to **System > User management**, and then select **Users, Groups, Roles** and **Permissions** (non-editable) to display the corresponding overview.

Whitelist URLs

The platform needs to access external data sources to ingest intel, as well as to enrich entities and observables. You may want to whitelist these URLs, domains and addresses, so that the platform can communicate with the external intel and service providers.

Repositories

When installing or upgrading the platform and its dependencies, the system needs to access the following source repositories.

Repository URL	Belongs to	Repo type
https://downloads.eclecticiq.com/platform	EclecticIQ Platform	deb, rpm
https://dl.fedoraproject.org/pub/epel/7/x86_64/	EPEL	rpm
https://centos7.iuscommunity.org/	IUS	rpm
http://download.oracle.com/	Oracle Java JDK	deb, rpm
https://nginx.org/packages/ubuntu/pool/nginx/n/nginx/	Nginx	deb
https://nginx.org/packages/centos/7/x86_64/	Nginx	rpm
https://download.postgresql.org/pub/repos/apt/pool/9.5/	PostgreSQL	deb
https://download.postgresql.org/pub/repos/yum/9.5/redhat/	PostgreSQL	rpm
https://packages.elastic.co/	ELK stack	deb, rpm
http://debian.neo4j.org/	Neo4j	deb
http://yum.neo4j.org/	Neo4j	rpm
http://download.redis.io/releases/	Redis	deb, rpm
https://deb.nodesource.com/	Node.js	deb
https://rpm.nodesource.com/	Node.js	rpm

Enrichers and feeds

Feeds and enrichers access data sources through these URLs. Whitelist the domains and allow traffic to and from them.

Domain	Belongs to	Type
<a href="http://<elasticsearch_url>:9200/<schema_resource>">http://<elasticsearch_url>:9200/<schema_resource>	Elasticsearch sightings	enricher/API

Domain	Belongs to	Type
https://cybercrime-portal.fox-it.com/	Fox-IT InTELL Portal	enricher/API, incoming feed/API
https://api.intel471.com/v1/	Intel 471	enricher/API, incoming feed/API
http://api.openresolve.com/{}/{}	OpenDNS OpenResolve	enricher/API
http://10.0.1.60:8000/ (example)	PyDat	enricher/API
https://stat.ripe.net/data/geoloc/data.json?resource={IP_address}	RIPEstat GeolIP	enricher/API
https://stat.ripe.net/data/whois/data.json?resource={IP_address}	RIPEstat Whois	enricher/API
https://panacea.threatgrid.com/api/v2/	Cisco Threat Grid	enricher/API, incoming feed/API
https://www.virustotal.com/vtapi/v2/{}	VirusTotal	enricher/API
https://endlesstunnel.info/v3	Flashpoint AggregINT	enricher/API
https://endlesstunnel.info/v3	Flashpoint Blueprint	enricher/API
https://endlesstunnel.info/v3	Flashpoint Thresher	enricher/API
https://api.passivetotal.org/v2	PassiveTotal Whois	enricher/API
https://api.passivetotal.org/v2	PassiveTotal Passive DNS	enricher/API
https://api.passivetotal.org/v2	PassiveTotal IP/Domain	enricher/API
https://api.passivetotal.org/v2	PassiveTotal Malware	enricher/API
http://10.0.1.22:8089/ (example)	Splunk sightings	enricher/API
http://api.domaintools.com/v1/{}/host-domains	DomainTools Hosted Domains	enricher
http://api.domaintools.com/v1/reputation	DomainTools Reputation	enricher
https://api.domaintools.com/v1/{}/host-domains	DomainTools Suspicious Domains	enricher
https://api.isightpartners.com/search/{}	FireEye iSIGHT	enricher
https://app.recordedfuture.com/live/sc/entity/{}	Recorded Future	enricher
https://unshorten.me/s/{}	Unshorten-URL	enricher
https://api.dnsdb.info/{}	Farsight DNSDB	enricher
https://www.threatcrowd.org/{}	ThreatCrowd	enricher
https://censys.io/api/v1/search/ipv4	Censys	enricher
http://api.domaintools.com/v1/{}/name-server-domains/	DomainTools Malicious Server Domains	enricher

Domain	Belongs to	Type
http://api.domaintools.com/v1/{}/whois/parsed	DomainTools Retrieve Parsed Whois Observables	enricher
https://intelapi.crowdstrike.com/indicator/v1/search/{}	CrowdStrike Falcon Intelligence Indicator	enricher

Other URLs

Domain	Belongs to	Type
http://<variable_subdomain>.cyberfeed.net:<port_number>	AnubisNetworks	incoming feed
https://api.threatrecon.co/	Threat Recon	incoming feed
http://hailataxii.com	Hail a TAXII	open source cyber threat intelligence source
https://test.taxiistand.com/	TAXII Stand	public OpenTAXII test server

Open ports

The platform components communicate with the platform and with each other through these ports. Make sure they are open within the platform network.

Port	Belongs to
9200	elasticsearch
5601	kibana
7474	neo4j
4008	neo4j-batching
8008	platform-api
5432	postgresql-9.5
6379	redis
6755	logstash
80; 443	nginx
25; 587	postfix
9001	supervisor

Configure the platform

Configure the third-party products the platform interacts with, and then update the platform settings to reflect the configuration changes.

After installing the platform, you can proceed to configuring it, along with the required third-party components such as web server, graph, indexing, and search:

- Configure the database (step 2 of 2) (PostgreSQL)
- Configure the web server (Nginx)
- Configure the data structure store (Redis)
- Configure the search server (Elasticsearch)
- Configure the log aggregator (Logstash)
- Configure the graph database (Neo4j)
- Update the platform settings.

To successfully execute several commands in the command line or in the terminal, you may need root-level access rights.

To obtain admin rights, run the following command(s):

```
$ sudo su -
```

Alternatively:

- Grant admin rights to a specific user, who can then log in with their password to perform admin tasks:

```
$ su - {user_name}
```

Or:

- Prefix `sudo` to the command you want to run:

```
$ sudo {command}
```

Configure PostgreSQL (2 of 2)



Warning:

You should configure PostgreSQL in 2 steps:

- Create the main data storage *before* installing the platform.
- Add the database to the platform settings *after* installing the platform.

Add the database to the platform settings

- In the `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` platform configuration file, update the `SQLALCHEMY_DATABASE_URI` parameter with the designated database URI for your environment:

```
# Format:
SQLALCHEMY_DATABASE_URI = "postgresql://<username>:<password>@<host>:<port>/<database-name>"

# Example:
SQLALCHEMY_DATABASE_URI = "postgresql://test:test@localhost:5432/platform"
```

username	Replace this placeholder with the user name of the user that can access the database. Example: <code>test</code>
password	Replace this placeholder with the corresponding user password. Example: <code>test</code>
host	Replace this placeholder with the host name identifying the location where the database is hosted. Example: <code>localhost</code>
port	The database access port. Default value: <code>5432</code>
database-name	The assigned name to identify the database. Example: <code>platform</code>

Enable the service

- Enable the PostgreSQL service to start at system bootup:

```
$ systemctl enable postgresql
```

- Restart the PostgreSQL service:

```
$ systemctl restart postgresql
```

- To check if PostgreSQL is running, run the following command(s):

```
$ systemctl status postgresql
```

- Verify that user and database are successfully created:

```
$ psql -U test -d platform -W
```

Configure Nginx

Nginx (<http://nginx.org/en/download.html>) is the web server we are setting up for the platform.

To configure Nginx, do the following:

- Go to `/etc/nginx`:

```
$ cd /etc/nginx
```

- Back up `/etc/nginx/nginx.conf`, and replace it with the reference `nginx.conf` file shipped with the platform:

```
$ cp /opt/eclecticiq/etc-extras/nginx/nginx.conf /etc/nginx/nginx.conf
```

- Edit `/etc/nginx/nginx.conf` to reflect your actual configuration:

```
$ nano /etc/nginx/nginx.conf
```

Set Nginx user and group

Check the following parameters and their settings, since they affect interoperability between the platform and Nginx:

- The `user` should have the default Nginx values of `www-data` for both user and group name:

```
user      www-data www-data;
```

Check access log and log format

- Verify the Nginx **access log** (<https://www.nginx.com/resources/admin-guide/logging-and-monitoring/>) format to make sure Nginx can recognize and consume the *expected format for web server logs*. The `access.log` file location is `/var/log/nginx/`.
- In the `nginx.conf` file, make sure that the `log_format` directive holds the configuration parameters for the web server **log format** (https://nginx.org/en/docs/http/nginx_http_log_module.html#log_format). The `nginx.conf` file location is `/opt/eclecticiq/etc-extras/nginx/`. The following example shows the default `log_format` section in the Nginx `nginx.conf` file, as per Nginx 1.8:

```
# Recommended basic Nginx configuration from EclecticIQ

user          www-data www-data;
error_log     /var/log/nginx/error.log info;
pid           /run/nginx.pid;
worker_processes auto;

events {
    worker_connections 1024;
}

http {
    server_tokens off;
    include mime.types;
    default_type application/octet-stream;

    log_format json '{'
        '"remote_addr": "$remote_addr",'
        '"remote_user": "$remote_user",'
        '"time_local": "$time_local",'
        '"request": "$request",'
        '"status": $status,'
        '"body_bytes_sent": $body_bytes_sent,'
        '"http_referer": "$http_referer",'
        '"http_user_agent": "$http_user_agent",'
        '"http_x_forwarded_for": "$http_x_forwarded_for"'
        '}'

    access_log /var/log/nginx/access.log json;
    include /etc/nginx/conf.d/*.conf;
}
```

- Make sure the **access_log** (https://nginx.org/en/docs/http/nginx_http_log_module.html#access_log) format value matches the corresponding value specified in `log_format <format_name>`. The default Nginx log format for EclecticIQ Platform is `json`:

```
access_log /var/log/nginx/access.log json;
```

- To enable Nginx to pick up and start the platform configuration, copy *platform.conf*
 - from `/opt/eclecticiq/etc/nginx/conf.d/platform.conf`
 - to `/etc/nginx/conf.d`

```
$ cp /opt/eclecticiq/etc/nginx/conf.d/platform.conf /etc/nginx/conf.d/platform.conf
```

- Open *platform.conf* in a text editor:

```
$ nano /etc/nginx/conf.d/platform.conf
```

- Look for the `server_name` property.
- Replace `server_name <default_server_name>;` with the appropriate platform host name for your environment:

```
// Default server name value:
server_name <default_server_name>;

// Change it to your platform host name:
server_name <platform_server_name>;
```

- Make sure that the SSL certificate paths point to the correct locations where your certificates are stored.
Example:

```
ssl_certificate      /etc/ssl/certs/<certificate_name>.cert;
ssl_certificate_key  /etc/ssl/private/<key_name>.key;
```

Enable the service

- Enable the Nginx service to start at system bootup:

```
$ systemctl enable nginx
```

- Start the Nginx web server:

```
$ systemctl start nginx
```

- Verify that Nginx is up and running by checking the web server status:

```
$ systemctl status nginx
```

Configure Redis

To configure Redis, do the following:

- Go to */etc/redis*:

```
$ cd /etc/redis
```

- Back up */etc/redis/redis.conf*, and replace it with the reference *redis.conf* file shipped with the platform:

```
$ cp /opt/eclecticiq/etc-extras/redis.conf /etc/redis/redis.conf
```

For reference, look up a copy of the **self-documented file**

(<https://raw.githubusercontent.com/antirez/redis/2.8/redis.conf>), and the **Redis configuration documentation** (<https://redis.io/topics/config>).

Check the Redis configuration

- Edit `/etc/redis/redis.conf` to reflect your actual configuration.
- Check the following parameters and their settings, since they affect interoperability between the platform and Redis:
 - `port 6379`: this is the default port for Redis.
 - `dbfilename dump.rdb`: the default file name of the database Redis saves snapshots to.
 - `dir /<path_to_dir>/<redis_snapshot>`: sets the location where Redis stores the snapshot database.

Check the Redis system configuration

- Check, and if necessary edit, `/etc/systemd/system/redis.service` to reflect your actual configuration. This is the system configuration file that enables Systemd to manage Redis tasks and processes.



Warning: Redis will fail to start and run normally if the parameters in this file are not set correctly for the target environment.

Check permissions

Make sure the target directory to save application data to exists before setting it in the configuration file, and that it is accessible to read and write data.

If necessary, do the following:

- Create the target directory:

```
$ mkdir /<path_to_dir>/<target_dir>
```

- Make sure the correct user owns the target directory to access it in read and write modes.

```
$ chown -R redis:redis /media/redis
```

`redis:redis` should own also the following locations:

```
$ chown -R redis:redis /var/lib/redis
```

```
$ chown -R redis:redis /var/log/redis
```

Check the platform settings

- Open `/opt/eclecticiq/etc/eclecticiq/platform_settings.py`, and verify that `REDIS_URL` points to the correct location in your environment:

```
REDIS_URL="redis://<redis.host>:<redis.port>"
```

- To verify that Redis is correctly installed, do the following:

```
$ type redis-server
```

```
$ type redis-cli
```

Enable the service

- Enable the Redis service to start at system bootup:

```
$ systemctl enable redis
```

- To start Redis, run the following command(s):

```
$ systemctl start redis
```

- To check if Redis is running, run the following command(s):

```
$ systemctl status redis
```

- To verify that Redis is working properly, run the following command(s):

```
# Launch redis-cli
$ redis-cli

# Ping Redis to ask how it is doing
$ > ping

# Redis responds
PONG
```

Configure Elasticsearch

To configure Elasticsearch, do the following:

- Go to */etc/elasticsearch*:

```
$ cd /etc/elasticsearch
```

- Back up */etc/elasticsearch/elasticsearch.yml*, and replace it with the reference *elasticsearch.yml* file shipped with the platform:

```
$ cp /opt/eclecticiq/etc-extras/elasticsearch/elasticsearch.yml
/etc/elasticsearch/elasticsearch.yml
```

- Edit */etc/elasticsearch/elasticsearch.yml* to reflect your actual configuration:

```
$ nano /etc/elasticsearch/elasticsearch.yml
```


Check the following parameters and their settings, since they affect interoperability between the platform and Elasticsearch:

- Set `path.data` to the location where Elasticsearch stores its data.

Check permissions

Make sure the target directory to save application data to exists before setting it in the configuration file, and that it is accessible to read and write data.

If necessary, do the following:

- Create the target directory:

```
$ mkdir /<path_to_dir>/<target_dir>
```

- Make sure the correct user owns the target directory to access it in read and write modes.

```
$ chown -R elasticsearch:elasticsearch /media/elasticsearch
```

Disable dynamic scripting

- You should disable **dynamic scripting**

(<https://www.elastic.co/guide/en/elasticsearch/reference/current/modules-scripting.html#enable-dynamic-scripting>) by either removing the `script.inline` property from the configuration file, or by setting it to `false`:

```
script.inline: false
script.indexed: true
```

Example *elasticsearch.yml* file for reference:

```
config:

  cluster.name: intel

  discovery.zen.ping.multicast: false

  index.number_of_replicas: 0

  index.number_of_shards: 1

  node.local: true

  path:

    data: /<path_to>/<elasticsearch_index_datastore>

  script.indexed: true

  script.inline: false
```

Set heap size and custom tmp dir

You should allocate enough memory to the **heap size**

(<https://www.elastic.co/guide/en/elasticsearch/guide/current/heap-sizing.html>), based on your system configuration and the size of the Elasticsearch index.

The minimum recommended value is 2 GB: `ES_HEAP_SIZE=2g`.

You can set this option in the Elasticsearch configuration file:

- Go to `/etc/default` and open the `elasticsearch` configuration file:

```
$ nano /etc/default/elasticsearch
```

Example `elasticsearch` file for reference:

```
ES_HOME=/usr/share/elasticsearch
ES_HEAP_SIZE=2g
ES_JAVA_OPTS="-Djna.tmpdir=/<path_to>/<elasticsearch_tmp_dir>"
MAX_OPEN_FILES=65535
MAX_MAP_COUNT=262144
LOG_DIR=/var/log/elasticsearch
DATA_DIR=/media/elasticsearch
WORK_DIR=/tmp/elasticsearch
CONF_DIR=/etc/elasticsearch
ES_USER=elasticsearch
```

- Make sure you set the Elasticsearch heap size to at least 2 GB or more, if your system allows it:

```
ES_HEAP_SIZE=2g
```

- Optionally, you may **change the default temp directory**

(<https://github.com/elastic/elasticsearch/issues/18406>) and set a custom temporary directory, if the current one is mounted with the `noexec` option.

To do so, **edit or add** (<https://www.elastic.co/guide/en/elasticsearch/reference/2.4/setup-configuration.html#setup-configuration-memory>) the `ES_JAVA_OPTS` property:

```
ES_JAVA_OPTS="-Djna.tmpdir=/<path_to>/<elasticsearch_tmp_dir>"
```

- Verify that the `elasticsearch` user can read and write to the temp directory.

Check the Elasticsearch URL

- Open `/opt/eclecticiq/etc/eclecticiq/platform_settings.py`, and verify that `SEARCH_URL` points to the correct location corresponding to the Elasticsearch instance:

```
SEARCH_URL="http://<elasticsearch.host>:<elasticsearch.port>"
```

Enable the service

- Enable the Elasticsearch service to start at system bootup:

```
$ systemctl enable elasticsearch
```

- To start Elasticsearch, run the following command(s):

```
$ systemctl start elasticsearch
```

- To check if Elasticsearch is running, run the following command(s):

```
$ systemctl status elasticsearch
```

Configure Logstash

Logstash configuration files are copied automatically to the correct location during the post-install step. To verify that the configuration files exist in the target destination, run the following command(s):

```
$ ls -l /opt/eclecticiq/etc/logstash/conf.d/
```

These are the Logstash configuration files related to the platform:

```
filters.conf
input.conf
neo4j-batching.conf
opentaxii.conf
output.conf
platform-api.conf
platform-ui.conf
```

The following files control the **Logstash processing pipeline**

(<https://www.elastic.co/guide/en/logstash/current/pipeline.html>):

```
input.conf
filters.conf
output.conf
```

For further details, see the [how-to article on addressing logging issues in Kibana and Logstash configurations](#).

Enable the service

- Enable the Logstash service to start at system bootup:

```
$ systemctl enable logstash
```

- To start Logstash, run the following command(s):

```
$ systemctl start logstash
```

- To check if Logstash is running, run the following command(s):

```
$ systemctl status logstash
```

Configure Kibana

- To configure Kibana, simply replace `/opt/kibana/config/kibana.yml` with the provided `/opt/eclecticiq/etc-extras/kibana.yml`.
The default settings in the `/opt/eclecticiq/etc-extras/kibana.yml` configuration file should work in your environment without requiring any changes.

```
$ cp /opt/eclecticiq/etc-extras/kibana.yml /opt/kibana/config/kibana.yml
```

Kibana uses an index in Elasticsearch to store saved searches, visualizations, and dashboards.

The default property/value pair defining the index in the `kibana.yml` configuration file is `kibana_index: -kibana`.

The default parameters and values should not need any editing:

```
port: 5601
host: "0.0.0.0"
elasticsearch_url: "http://127.0.0.1:9200"
elasticsearch_preserve_host: true
kibana_index: "-kibana"
server.basePath: "/api/kibana"
default_app_id: "discover"
request_timeout: 300000
shard_timeout: 0
verify_ssl: false
bundled_plugin_ids:
  - plugins/dashboard/index
  - plugins/discover/index
  - plugins/doc/index
  - plugins/kibana/index
  - plugins/markdown_vis/index
  - plugins/metric_vis/index
  - plugins/settings/index
  - plugins/table_vis/index
  - plugins/vis_types/index
  - plugins/visualize/index
```

Kibana 4.5.x has an **issue** (<https://github.com/elastic/kibana/issues/6730>) that causes root owned files in `/opt/kibana/optimize/`. This can prevent Kibana from running.

It should be fixed in version 4.6.0.

To work around it, you can change permissions so that the `kibana` user owns or has read/write access to the `/opt/kibana/optimize/` directory.

To do so, run the following command(s):

```
$ chown -Rvf kibana:kibana /opt/kibana/optimize/*
```

Enable the service

- Enable the Kibana service to start at system bootup:

```
$ systemctl enable kibana
```

- To start Kibana, run the following command(s):

```
$ systemctl start kibana
```

- To check if Kibana is running, run the following command(s):

```
$ systemctl status kibana
```

Configure Neo4j

To configure Neo4j, do the following:

- Go to `/opt/eclecticiq/etc-extras/neo4j`.
- Copy the Neo4j configuration files shipped with the platform to `/etc/neo4j`, so that they replace the default configuration files created during the Neo4j installation:

```
$ cp /opt/eclecticiq/etc-extras/neo4j/* /etc/neo4j/
```

The action should copy the following Neo4j configuration files to the destination directory:

```
neo4j-server.properties  
neo4j-wrapper.conf  
neo4j.properties
```

Check Neo4j connectivity

Check the main connectivity properties to verify that Neo4j can communicate with other components.

Check neo4j-server.properties

- Open the `neo4j-server.properties` configuration file:

```
$ nano /etc/neo4j/neo4j-server.properties
```

- Edit the default property values, so that they reflect your system configuration:

```
org.neo4j.server.webserver.address=0.0.0.0
org.neo4j.server.webserver.port=7474
org.neo4j.server.database.location=/media/neo4j/graph.db
org.neo4j.server.webserver.https.enabled=false
dbms.security.auth_enabled=false
```

graph.db is the default name of the graph database file.

Make sure the directory where it is stored exists before setting it in this configuration file, and that Neo4j can read and write data to it.

Inspect the amount of allocated memory: Neo4j purrs [s](https://www.youtube.com/watch?v=drq_ww7ytzw) (https://www.youtube.com/watch?v=drq_ww7ytzw) when you give it plenty of memory to work with.

- Depending on your system resources, allocate at least 4 GB to Neo4j.
- Never set it below 1 GB.

To set the minimum and maximum memory values for Neo4j, do the following:

Check *neo4j-wrapper.conf*

- Open the *neo4j-wrapper.conf* configuration file:

```
$ nano /etc/neo4j/neo4j-wrapper.conf
```

- Make sure that the following properties are set to the corresponding values:

```
wrapper.java.initmemory=1024
wrapper.java.maxmemory=4096
```

- The recommended value for `wrapper.java.initmemory` is at least 4096 MB or more, if possible. The minimum value should never be lower than 1024 MB.
- The recommended value for `wrapper.java.maxmemory` is at least 4096 MB or more, if possible. This value should never be lower than `wrapper.java.initmemory`.

Check *neo4j.properties*

Set the size of the page cache to 5G:

- Open the *neo4j.properties* property file:

```
$ nano /etc/neo4j/neo4j.properties
```

- Make sure the `dbms.pagecache.memory` property is set to 5G:

```
dbms.pagecache.memory=5G
```

If this parameter is not configured, Neo4j defaults to using 50% of the available system RAM for page caching.

Check permissions

Make sure the target directory to save application data to exists before setting it in the configuration file, and that it is accessible to read and write data.

If necessary, do the following:

- Create the target directory:

```
$ mkdir /<path_to_dir>/<target_dir>
```

- Make sure the correct user owns the target directory to access it in read and write modes.
- The Neo4j installation package creates a `neo4j` user, but no group with the same name.
To assign file or directory ownership to this user, specify `nogroup` as the associated group name: `neo4j:nogroup`.

```
$ chown -R neo4j:nogroup /media/neo4j
```

Check the Neo4j URL

- Open `/opt/eclecticiq/etc/eclecticiq/platform_settings.py`, and verify that `NEO4J_URL` points to the correct Neo4j instance in your environment:

```
NEO4J_URL="http://<Neo4j.host>:<Neo4j.port>"
```

Enable the service

- Enable the Neo4j service to start at system bootup:

```
$ systemctl enable neo4j
```

- To start Neo4j, run the following command(s):

```
$ systemctl start neo4j
```

- To check if Neo4j is running, run the following command(s):

```
$ systemctl status neo4j
```

Configure Postfix

Postfix (<http://www.postfix.org/>) is the default **email server**

(https://en.wikipedia.org/wiki/message_transfer_agent) shipped with the platform. If you want to enable email-based platform features like email notifications, you need to configure Postfix to handle email traffic.

Postfix configuration and setup may vary, depending on the target environment the platform is installed on. The following links point to relevant sections of the official Postfix documentation:

- **Postfix documentation overview page** (<http://www.postfix.org/documentation.html>)
- **Postfix basic configuration** (http://www.postfix.org/basic_configuration_readme.html)
- **Postfix *main.cf* configuration file parameters** (<http://www.postfix.org/postconf.5.html>)

The platform ships with an example Postfix configuration file you can use as a customizable template: */opt/eclecticiq/etc-extras/postfix/main.cf*

- The default installation location of the Postfix configuration file is */etc/postfix*.
- The Postfix configuration file is *main.cf*.

To configure Postfix as the default email server for the platform do the following:

- Open *main.cf* in a text editor:

```
$ nano /etc/postfix/main.cf
```

Example *main.cf* file:

```
myhostname = box11.platform.host
mydomain = platform.host
myorigin = $mydomain
mydestination =

relayhost = [smtp.email.server.com]:587
inet_interfaces = loopback-only

smtp_sasl_security_options = noanonymous
smtp_fallback_relay = [relay.email.server.com]
smtp_sasl_auth_enable = yes
smtp_use_tls = yes
smtp_sasl_password_maps = hash:/etc/postfix/sasl_passwd
smtp_tls_policy_maps = hash:/etc/postfix/tls_policy
```

- **myhostname: specify the host name** (<http://www.postfix.org/postconf.5.html#myhostname>) of the platform host server where also Postfix is installed.
Example: `box11.platform.host`
- **mydomain: specify the internet name** (<http://www.postfix.org/postconf.5.html#mydomain>) of the platform host server where also Postfix is installed.
Example: `platform.host`
- **myorigin: specify the domain name** (<http://www.postfix.org/postconf.5.html#myorigin>) to use as sent-from and send-to address for local email traffic.
Automatic email messages generated by system processes such as cronjobs use the `myorigin` value for this purpose.
Example: `$mydomain` (domain with multiple machines), `$myhostname` (domain with one machine)
- **mydestination: specify the list of domains** (<http://www.postfix.org/postconf.5.html#mydestination>) the email server accepts email from.
Example: `localhost, localhost.$mydomain, mail.$mydomain, www.$mydomain, ftp.$mydomain`
- **relayhost: specify the next-hop destination** (<http://www.postfix.org/postconf.5.html#relayhost>) of non-local email. It identifies the next email handler in the chain of email servers handling non-local email traffic.
Format: `[smtp.domain.name]:port`
Example: `[smtp.email.server.com]:587`

- **inet_interfaces: specify the network interface addresses**
(http://www.postfix.org/postconf.5.html#inet_interfaces) Postfix receives mail on.
Example: all, loopback-only, 127.0.0.1
- **smtp_sasl_security_options: optionally specify one or more SASL security options**
(http://www.postfix.org/postconf.5.html#smtp_sasl_security_options) for the SMTP client.
Example: noplaintext, noactive, nodictionary, noanonymous, mutual_auth
- **smtp_fallback_relay: optionally specify one or more alternative relay hosts**
(http://www.postfix.org/postconf.5.html#smtp_fallback_relay) for SMTP destinations.
Example: relay.email.server.com
- **smtp_sasl_auth_enable: enable SASL authentication**
(http://www.postfix.org/postconf.5.html#smtp_sasl_auth_enable).
Example: yes, no
- **smtp_use_tls: enable using TLS** (http://www.postfix.org/postconf.5.html#smtp_use_tls) when available on the remote server.
Example: yes, no
- **smtp_sasl_password_maps: optionally specify a lookup table with one or more username:password entries**
(http://www.postfix.org/postconf.5.html#smtp_sasl_password_maps) per row to identify sender, remote host name, or next-hop domain.
Example: username:password
- **smtp_tls_policy_maps: optionally specify a lookup table with one or more SMTP client TLS security policies**
(http://www.postfix.org/postconf.5.html#smtp_sasl_password_maps) by next-hop destination.
Example: [smtp.email.server.com]:587 encrypt

Enable the service

- Enable the Postfix service to start at system bootup:

```
$ systemctl enable postfix
```

- To start Postfix, run the following command(s):

```
$ systemctl start postfix
```

- To check if Postfix is running, run the following command(s):

```
$ systemctl status postfix
```

Update the platform settings

At this point you have set up dependencies and third-party components. You can now proceed to update the platform settings.

- Review the `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` platform settings file to make sure it reflects your environment.

The following example serves as a guideline:

```
PLATFORM_MANIFESTS_DIR = '/opt/eclecticiq/manifests'
COMPONENT_SHARED_SECRET = "<component_secret_key_value>"
SECRET_KEY = "<secret_key_value>"
PROXY_URL_FILE_PATH = '/opt/eclecticiq/etc/eclecticiq/proxy_url'

SUPERVISORD_HOSTS = '127.0.0.1:9001'
SYSTEMD_SERVICES = [
    'elasticsearch',
    'logstash',
    'postfix',
    'neo4j',
    'postgresql-9.5',
    'redis',
]

SQLALCHEMY_DATABASE_URI = "postgresql://<username>:<password>@localhost:5432/platform"
SQLALCHEMY_POOL_SIZE = 20
SQLALCHEMY_MAX_OVERFLOW = 20

DISCOVERY_WORKSPACE_SIZE_LIMIT = 500
DISCOVERY_SEARCH_LIMIT = 1000
DISCOVERY_RESULTS_LIMIT = 500

INGESTION_QUEUE_NAME = 'queue:ingestion:inbound'
ENRICHMENT_QUEUE_NAME = 'queue:enrichment:inbound'
GRAPH_QUEUE_NAME = 'queue:graph:inbound'
GRAPH_INGESTION_SPOOL_DIRECTORY = '/opt/eclecticiq/tmp/gi-storage'
SEARCH_QUEUE_NAME = 'queue:search:inbound'

REDIS_URL = 'redis://127.0.0.1:6379/'

KIBANA_URL = 'http://127.0.0.1:5601'

NEO4J_URL = 'http://127.0.0.1:7474'
NEO4J_BATCHING_URL = 'http://127.0.0.1:4008'
NEO4J_DEBUG = False

SEARCH_URL = 'http://127.0.0.1:9200'
OPENRESOLVE_URL = "http://api.openresolve.com/{}/{}"
RIPE_STAT_URL = "https://stat.ripe.net/data/{}/data.json?resource={}"
VIRUSTOTAL_API_SECRET = '<virustotal_api_secret_value>'
VIRUSTOTAL_API_URL = "https://www.virustotal.com/vtapi/v2/{}"

CELERY_QUEUES = [{"name": "providers", "routing_key": "eiq.providers.#"}, {"name": "analysis",
"routing_key": "eiq.analysis.#"}, {"name": "integrations", "routing_key": "eiq.integrations.#"},
{"name": "enrichers", "routing_key": "eiq.enrichers.#"}, {"name": "utilities", "routing_key":
"eiq.utilities.#"}, {"name": "priority_enrichers", "routing_key": "eiq.priority.enrichers.#"},
{"name": "priority_providers", "routing_key": "eiq.priority.providers.#"}, {"name":
"priority_utilities", "routing_key": "eiq.priority.utilities.#"}, {"name": "reindexing",
"routing_key": "eiq.reindexing.#"}]
```

**Warning:****Secret key and session token**

When you install, update or reinstall the platform, it is a good idea to change the following default values in the `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` configuration file:

- `SECRET_KEY = 'secret'`: change the secret key default value to a longer, random one.
- `JWT_EXPIRATION_DELTA = 60 * 30`: change the expiration time of the user authentication token as needed. The value is measured in seconds.

By default, the token expires 30 minutes after successfully signing in. The corresponding session is terminated, and you need to sign back in to the platform.

When human interaction is detected — for example, keystrokes or mouse activity — the token is automatically refreshed every 2 minutes. This prevents the system from signing out users who may be working or saving data at that time.

Therefore, the default maximum amount of idle time without any human interaction before being automatically signed out equals to *session token validity - 2 minutes*.

- When you set the spool directory for the graph ingestion in the platform configuration file, make sure the `eclecticiq` user can access it in write mode:

```
GRAPH_INGESTION_SPOOL_DIRECTORY = '/opt/eclecticiq/tmp/gi-storage' # 'eclecticiq' user needs
write rights to this dir
```

- Review the `/opt/eclecticiq/etc/eclecticiq/opentaxii.yml` OpenTAXII settings to make sure they reflect your environment.

Configure OpenTAXII

The `/opt/eclecticiq/etc/eclecticiq/opentaxii.yml` file contains the OpenTAXII configuration settings.

The following example serves as a guideline:

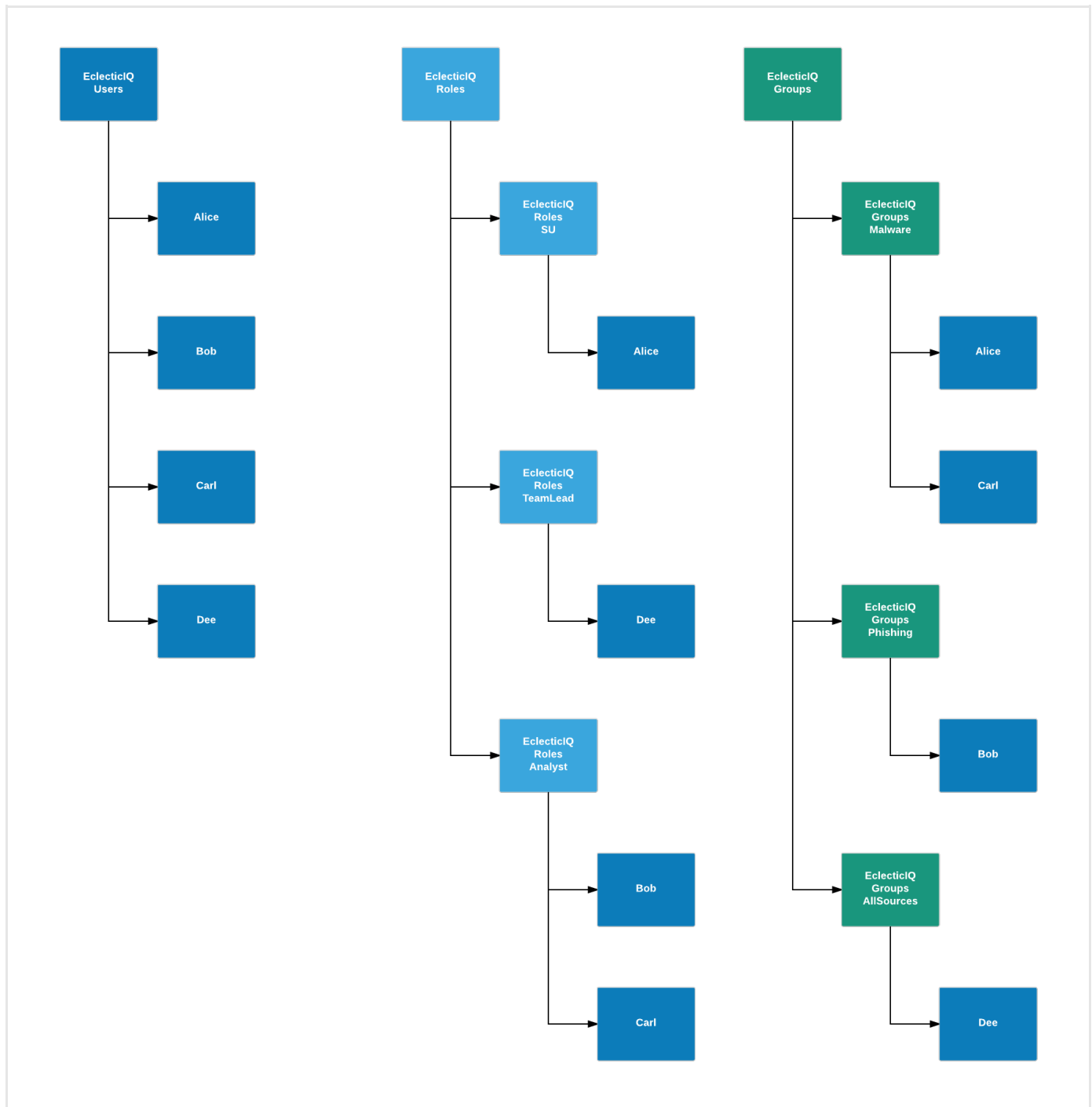
```
auth_api:
  class: eiq.opentaxii.auth.PlatformAuthAPI
  parameters: null
domain: <platform_host_name>
hooks: null
logging:
  opentaxii: debug
  root: debug
persistence_api:
  class: eiq.opentaxii.persistence.PlatformPersistenceAPI
  parameters: null
```

Configure LDAP authentication

You can configure the platform to import users and groups from an LDAP directory service. It is preferable not to mix LDAP users and local users in the platform. If you create local users in the platform, and then import LDAP users, LDAP users override the local ones. When such an override occurs, the platform recognizes LDAP users as internal, that is, local, and local platform users as external.

A standard way to structure LDAP groups can include the following steps:

- Create an LDAP group to hold groups, an LDAP group to hold users, and an LDAP group to hold roles.
 - Apply some basic standard naming rules for these LDAP groups. For example, use a prefix such as `EclecticIQ` or `EIQ`.
Example:
 - `EclecticIQUsers`
 - `EclecticIQRoles`
 - `EclecticIQGroups`
- Create LDAP *users* as necessary.
 - Assign these users to the `EclecticIQUsers` LDAP group.
- Create LDAP *roles* as necessary.
 - The role names you define in the LDAP structure should match exactly the existing role names in the platform.
 - Assign these roles to the `EclecticIQRoles` LDAP group.
- Create LDAP *groups* as necessary.
 - The group names you define in the LDAP structure should match exactly any existing group names in the platform.
 - Assign these groups to the `EclecticIQGroups` LDAP group.
- To define user group membership and user access policies, add the users to the child groups and the child roles of the `EclecticIQGroups` and `EclecticIQRoles` parent groups.



To configure LDAP authentication, do the following:

- Append the following attributes and set them to the appropriate values for your environment in the `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` platform settings file.
- Ensure you represent platform roles and groups as LDAP groups.
- LDAP role and group names, that is, the values of the `LDAP_ROLE_NAME_ATTR` and `LDAP_GROUP_NAME_ATTR` attributes, should exactly match the corresponding role and group name values in the platform.
- `LDAP_AUTH_ENABLED`: Boolean switch to enable/disable LDAP authentication.
Default value: `False` (LDAP disabled).
To enable LDAP authentication, set it to `True`.
- `LDAP_URI`: URI pointing to the designated LDAP instance. It supports `ldap://`, as well as `ldaps://` protocols.
Example:

```
LDAP_URI = "ldap://ldap.example.com"
```

- **LDAP_IGNORE_TLS_ERRORS:** Boolean switch to enable/disable TLS error checking such as invalid certificates, chain validation issues, and so on.
Default value: `True` (TLS errors are ignored).
- **LDAP_BIND_DN:** specify the valid credentials to bind to the LDAP connection, search for users, read groups and roles.
- **LDAP_BIND_PASSWORD:** specify the password associated to the binding credentials.
Example:

```
LDAP_BIND_DN = "cn=Manager,dc=ldap,dc=eclecticiq"  
LDAP_BIND_PASSWORD = "adminpassword"
```

- **LDAP_USERS_FILTER:** base and filter values used to search for user accounts.
The `{username}` placeholder value is replaced with an assigned user name.
Example:

```
LDAP_USERS_FILTER = (  
    "ou=EclecticIQUsers,dc=ldap,dc=eclecticiq",  
    "(uid={username})")
```

- **LDAP_GROUPS_FILTER:** base and filter values used to search for user groups.
The `{username}` placeholder value is replaced with an assigned user group name.
Example:

```
LDAP_GROUPS_FILTER = (  
    "ou=EclecticIQGroups,dc=ldap,dc=eclecticiq",  
    "(&(memberUid={username})(objectClass=posixGroup))")
```

- **LDAP_ROLES_FILTER:** base and filter values used to search for user roles.
The `{username}` placeholder value is replaced with an assigned role name.
Example:

```
LDAP_ROLES_FILTER = (  
    "ou=EclecticIQRoles,dc=ldap,dc=eclecticiq",  
    "(&(memberUid={username})(objectClass=posixGroup))")
```

- **LDAP_USER_FIRSTNAME_ATTR:** define the naming attribute
(<http://ldapwiki.com/wiki/best%20practices%20for%20ldap%20naming%20attributes>) used to extract a user's first/given name.
- **LDAP_USER_LASTNAME_ATTR:** define the naming attribute used to extract a user's surname/family name.
- **LDAP_USER_EMAIL_ATTR:** define the naming attribute used to extract a user's email address.
- **LDAP_ROLE_NAME_ATTR:** define the naming attribute used to extract a user's role details.
- **LDAP_GROUP_NAME_ATTR:** define the naming attribute used to extract a user's group details.
- **LDAP_USER_IS_ADMIN_ATTR:** specify if the user should be granted admin rights.
To grant a user admin rights, assign the attribute the following value: `'isEclecticIQAdmin'`.
Example:

```
LDAP_USER_IS_ADMIN_ATTR = 'isEclecticIQAdmin'
```

Example LDAP configuration:

```
# LDAP settings

LDAP_AUTH_ENABLED = True
LDAP_URI = "ldaps://10.0.2.212"
LDAP_IGNORE_TLS_ERRORS = True

LDAP_BIND_DN ="cn=Manager,dc=ldap,dc=eclecticiq"
LDAP_BIND_PASSWORD ="adminpassword"

LDAP_USERS_FILTER = (
    "ou=EclecticIQUsers,dc=ldap,dc=eclecticiq",
    "(uid={username})")

LDAP_GROUPS_FILTER = (
    "ou=EclecticIQGroups,dc=ldap,dc=eclecticiq",
    "(&(memberUid={username})(objectClass=posixGroup))")

LDAP_ROLES_FILTER = (
    "ou=EclecticIQRoles,dc=ldap,dc=eclecticiq",
    "(&(memberUid={username})(objectClass=posixGroup))")

LDAP_USER_FIRSTNAME_ATTR = "cn"
LDAP_USER_LASTNAME_ATTR = "sn"
LDAP_USER_EMAIL_ATTR = "mail"

LDAP_ROLE_NAME_ATTR = "cn"
LDAP_GROUP_NAME_ATTR = "cn"
LDAP_USER_IS_ADMIN_ATTR = "isEclecticIQAdmin"
```



LDAP referrals (<http://www.openldap.org/doc/admin24/referrals.html>) are not supported.

Configure LDAP to work with AD

You can set up LDAP authentication to work with AD (Active Directory).

EclecticIQ Platform provides a generic AD implementation. You can use it as a template to fine-tune attributes and parameters to suit the specific AD setup in your environment.

This table sums up the main differences between a vanilla LDAP configuration, and an LDAP setup that enables interoperability with AD:

LDAP	AD
-	memberOf:1.2.840.113556.1.4.1941
objectClass=posixGroup	objectClass=group
memberUid={username}	member={user_dn}

LDAP	AD
cn	cn, sAMAccountName, givenName

- `memberOf:1.2.840.113556.1.4.1941`: this string enables recursive filtering and match search. The magic number is an **OID** (<http://www.oid-info.com/cgi-bin/display?oid=1.2.840.113556.1.4.1941&action=display>) that identifies the **LDAP_MATCHING_RULE_IN_CHAIN** ([https://msdn.microsoft.com/en-us/library/windows/desktop/aa746475\(v=vs.85\).aspx](https://msdn.microsoft.com/en-us/library/windows/desktop/aa746475(v=vs.85).aspx)) matching rule. You need to include this string if you want to enable recursive pattern search inside a hierarchical data structure.
- `objectClass=group`: the group name that identifies AD groups, as opposed to `objectClass=posixGroup`, which represents Unix groups.
- `member={user_dn}: {user_dn}` takes the returned user object value. This is the full user **DN** (<http://ldapwiki.com/wiki/distinguished%20names>) that is filled after the first user-search operation.
- `cn, sAMAccountName, givenName`: naming attributes that can vary, depending on the specific AD setup in your environment.

Example

To successfully execute several commands in the command line or in the terminal, you may need root-level access rights.

To obtain admin rights, run the following command(s):

```
$ sudo su -
```

Alternatively:

- Grant admin rights to a specific user, who can then log in with their password to perform admin tasks:

```
$ su - {user_name}
```

Or:

- Prefix `sudo` to the command you want to run:

```
$ sudo {command}
```

- Append the following parameters to `/opt/eclecticiq/etc/eclecticiq/platform_settings.py`.
- Restart the `platform-api` service:

```
$ supervisorctl restart platform-api
```

LDAP/AD configuration enabling users to sign in with their designated user name (example: `k_soze`)

`LDAP_USERS_FILTER = "sAMAccountName = {username}"` sets signing in with the user's user name.


```

LDAP_AUTH_ENABLED = True
LDAP_URI = 'ldap://10.0.12.154'

LDAP_BIND_DN = "admin@example.com"
LDAP_BIND_PASSWORD = "=BKHpUW3f="

LDAP_USERS_FILTER = (
    "cn=Users,dc=eiq,dc=local",
    "sAMAccountName={username}")

LDAP_GROUPS_FILTER = (
    "cn=Users,dc=eiq,dc=local",
    "(&(memberOf:1.2.840.113556.1.4.1941:=cn=EclecticIQGroups,cn=Users,dc=eiq,dc=local) "
    "(objectClass=group) "
    "(member={user_dn}))")

LDAP_ROLES_FILTER = (
    "cn=Users,dc=eiq,dc=local",
    "(&(memberOf:1.2.840.113556.1.4.1941:=cn=EclecticIQRoles,cn=Users,dc=eiq,dc=local) "
    "(objectClass=group) "
    "(member={user_dn}))")

LDAP_USER_FIRSTNAME_ATTR = 'givenName'
LDAP_USER_LASTNAME_ATTR = 'sn'
LDAP_USER_EMAIL_ATTR = 'mail'
LDAP_ROLE_NAME_ATTR = 'sAMAccountName'
LDAP_GROUP_NAME_ATTR = 'sAMAccountName'

```

LDAP/AD configuration enabling users to sign in with their full name (example: *Keyser Söze*)

LDAP_USERS_FILTER = "cn={username}" sets signing in with the user's full name.

```

LDAP_AUTH_ENABLED = True
LDAP_URI = 'ldap://10.0.12.154'

LDAP_BIND_DN = "admin@example.com"
LDAP_BIND_PASSWORD = "=BKHpUW3f="

LDAP_USERS_FILTER = (
    "cn=Users,dc=eiq,dc=local",
    "cn={username}")

LDAP_GROUPS_FILTER = (
    "cn=Users,dc=eiq,dc=local",
    "(&(memberOf:1.2.840.113556.1.4.1941:=cn=EclecticIQGroups,cn=Users,dc=eiq,dc=local) "
    "(objectClass=group) "
    "(member={user_dn}))")

LDAP_ROLES_FILTER = (
    "cn=Users,dc=eiq,dc=local",
    "(&(memberOf:1.2.840.113556.1.4.1941:=cn=EclecticIQRoles,cn=Users,dc=eiq,dc=local) "
    "(objectClass=group) "
    "(member={user_dn}))")

LDAP_USER_FIRSTNAME_ATTR = 'givenName'
LDAP_USER_LASTNAME_ATTR = 'sn'
LDAP_USER_EMAIL_ATTR = 'mail'
LDAP_ROLE_NAME_ATTR = 'sAMAccountName'
LDAP_GROUP_NAME_ATTR = 'sAMAccountName'

```

Configure SAML authentication

You can configure the platform to authenticate users with **SAML** (<https://www.oasis-open.org/standards#samlv2.0>). **SAML** (<https://docs.oasis-open.org/security/saml/v2.0/>) is an XML-based format to exchange authentication and authorization data, usually between an identity provider and a service provider.

It is preferable not to mix SAML users and local users in the platform, as SAML-authenticated users can override local platform users. When such an override occurs, the platform recognizes SAML users as internal, that is, local, and local platform users as external.

The current SAML implementation in the platform supports SAML authentication and authorization, but not SSO.

This is the **SAML flow implemented in the platform**

(https://en.wikipedia.org/wiki/saml_2.0#sp_redirect_request.3b_idp_post_response):

- The user makes a SAML sign in request to the platform.
- The platform acts as the service provider, and it redirects the user to the identity provider.
- The identity provider identifies the user, and it issues a document back to the user.
- The user makes a request to the service provider to pass a token, and then the user requests the target resource (for example, a web page).
- The service provider delivers the requested resource.

To implement SAML authentication, you need to:

- Set up and configure a SAML identity provider.
- Configure the platform to perform authentication through SAML as a service provider.
- Generate a metadata XML file, so that the identity provider can recognize the platform instance as a legitimate service provider.

To configure SAML authentication in the platform, do the following:

- Append the following attributes and set them to the appropriate values for your environment in the `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` platform settings file.
- SAML role and group names, that is, the values of the `SAML_USER_ROLES_ATTR` and `SAML_USER_GROUPS_ATTR` attributes, should exactly match the corresponding role and group name values in the platform.
- `SAML_AUTH_ENABLED`: Boolean switch to enable/disable authentication.
Default value: `False` (SAML disabled).
To enable SAML authentication, set it to `True`.
- `SAML_IDP_METADATA`: it can contain *either* a `url` or a `file` parameter (not both):
 - `url`: a valid URL pointing to the XML metadata the identity provider needs to access to recognize the platform instance as a legitimate service provider.
 - `file`: a valid path pointing to the XML metadata file the identity provider needs to access to recognize the platform instance as a legitimate service provider.
- `SAML_IDP_ENTITYID`: define a URI that uniquely identifies the configured SAML identity provider.
The URI should not exceed 1024 characters. The SAML 2.0 core specification recommends that a system entity use a URL containing its own domain name to identify itself.
- `SAML_USER_USERID_ATTR`: define the naming attribute used to extract a user ID.
- `SAML_USER_EMAIL_ATTR`: define the naming attribute used to extract a user's email address.
- `SAML_USER_GROUPS_ATTR`: define the naming attribute used to extract a user's group details.

- **SAML_USER_ROLES_ATTR**: define the naming attribute used to extract a user's role details.
- **SAML_USER_FIRSTNAME_ATTR**: define the naming attribute used to extract a user's first/given name.
- **SAML_USER_LASTNAME_ATTR**: define the naming attribute used to extract a user's surname/family name.
- **SAML_USER_IS_ADMIN_ATTR**: specify if the user should be granted admin rights.
To grant a user admin rights, assign the attribute the following value: "isEclecticIQAdmin".
Example:

```
SAML_USER_IS_ADMIN_ATTR = "isEclecticIQAdmin"
```

- **SAML_SIGN_AUTHN_REQ**: Boolean switch to enable/disable mandatory digital signing for authentication requests.
Default value: `False` (authentication request signing disabled)
- **SAML_WANT_ASSERT_SIGNED**: Boolean switch to enable/disable mandatory digital signing for assertions received from the identity provider.
Default value: `True` (assertions must be signed)
- **SAML_WANT_RESPONSE_SIGNED**: Boolean switch to enable/disable mandatory digital signing for responses received from the identity provider.
Default value: `False` (response signing disabled)
- **SAML_ENC_KEY**: define the path to the location where the decryption key is stored.
The key should include a `Recipient` attribute defining the entity the key was encrypted for.
The value of the `Recipient` attribute should be the URI identifier of a SAML system entity.
- **SAML_ENC_CERT**: define the path to the location where the decryption certificate is stored.
- **SAML_XMLSEC_BIN**: define the path to the local **xmlsec** (<https://pypi.python.org/pypi/xmlsec>) installation.
xmlsec provides Python bindings for the XML Security Library.
- **SAML_METADATA_ORG**: define metadata information to identify the platform instance or the organization.
 - **name**: define one or more names to identify the platform instance or the organization, along with the corresponding language identifier(s).
Names you define here may or may not be suitable for human consumption.
 - **display_name**: define one or more names to identify the platform instance or the organization, along with the corresponding language identifier(s).
Names you define here are suitable for human consumption.
 - **url**: define one or more URIs pointing to resources users can look up for additional information.
For example, a corporate web site, a blog, a publicly accessible wiki page, and so on.
- **SAML_METADATA_CONTACT_PERSON**: define metadata information to identify a designated contact person for the platform instance or the organization.
 - **given_name**: define the contact person's first/given name.
 - **sur_name**: define the contact person's surname/family name.
 - **email_address**: define the contact person's email address.
 - **contact_type**: define the contact person's professional role.
For example, **system administrator** (<https://xkcd.com/705/>), **security officer** (<https://xkcd.com/327/>), **software engineer** (<https://xkcd.com/378/>), and so on.

Example SAML configuration:

```
SAML_AUTH_ENABLED = False

SAML_IDP_METADATA = {
    'url': 'https://idp.testshib.org/idp/shibboleth',
    # 'file': '/idp-metadata.xml'
}

SAML_IDP_ENTITYID = "https://idp.testshib.org/idp/shibboleth"

# required attributes
SAML_USER_USERID_ATTR = 'uid'
SAML_USER_EMAIL_ATTR = 'email'
SAML_USER_GROUPS_ATTR = 'EclecticIQGroups'
SAML_USER_ROLES_ATTR = 'EclecticIQRoles'

# optional attributes
SAML_USER_FIRSTNAME_ATTR = 'givenName'
SAML_USER_LASTNAME_ATTR = 'sn'
SAML_USER_IS_ADMIN_ATTR = 'isEclecticIQAdmin'

SAML_SIGN_AUTHN_REQ = False
SAML_WANT_ASSERT_SIGNED = True
SAML_WANT_RESPONSE_SIGNED = False

SAML_ENC_KEY = '/tmp/saml-keys/platform-saml-key.pem'
SAML_ENC_CERT = '/tmp/saml-certs/platform-saml-cert.crt'

SAML_XMLSEC_BIN = '/usr/bin/xmlsec1'

SAML_METADATA_ORG = {
    'name': 'Gus Meth Lab',
    'display_name': [('Los Pollos Hermanos', 'es-mx')],
    'url': 'http://lospolloshermanos.com',
}

SAML_METADATA_CONTACT_PERSON = {
    'given_name': 'Gus',
    'sur_name': 'Fring',
    'email_address': ['gus.fring@lospolloshermanos.com'],
    'contact_type': 'General manager',
}
```

Test SAML authentication

To test SAML authentication in the platform, you can use the following example as a guideline.

To successfully execute several commands in the command line or in the terminal, you may need root-level access rights.

To obtain admin rights, run the following command(s):

```
$ sudo su -
```

Alternatively:

- Grant admin rights to a specific user, who can then log in with their password to perform admin tasks:

```
$ su - {user_name}
```

Or:

- Prefix `sudo` to the command you want to run:

```
$ sudo {command}
```

- Open the `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` platform settings file, and enable SAML authentication.
- Generate an SSL key/certificate pair.
You can use **this handy Bash script** (<https://gist.github.com/adrianwebb/3313395>) to do that (requires **OpenSSL** (<https://www.openssl.org/>)).
- Save the generated files to a directory.
- Specify the correct paths to the key and the certificate in the `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` platform settings file:

```
SAML_AUTH_ENABLED = True
SAML_ENC_KEY = '/tmp/saml-keys/platform-saml-key.pem'
SAML_ENC_CERT = '/tmp/saml-certs/platform-saml-cert.crt'
```

- Restart the `platform-api` service:

```
$ supervisorctl restart platform-api
```

- Go to the platform sign in page, and verify that the page includes the option to sign in with SAML.
- Generate the SAML service provider metadata, which you need to register the platform instance with the designated identity provider:
 - Log in to the platform host system.
 - Change user to `eclecticiq`.
 - As `eclecticiq` user, run the following command(s):

```
$ . /opt/eclecticiq/platform/api/bin/activate
(api) $ export INTELWORKS_PLATFORM_SETTINGS="/opt/eclecticiq/etc/eclecticiq/platform_settings.py"
(api) $ manage generate_saml_metadata_xml -o /tmp/saml-serviceprovider-metadata.xml
```

- Go to the save-to directory — `/tmp` — in the example and verify that the XML metadata file exists.
- Upload, send or transmit the metadata file to the configured SAML identity provider.
This may vary, depending on your SAML implementation, and on the configured SAML identity provider.
- Go to the platform sign in page, and verify that the page includes the option to sign in with SAML.
- Use the SAML option to sign in using valid SAML user credentials.
- You should be signed in with the user whose credentials you entered to authenticate, and you should be redirected to the platform dashboard.

Bootstrap the platform

As a last step after installation and configuration, bootstrap the platform and third-party components it interacts with, and then launch the platform to access it.

After installing and configuring EclecticIQ Platform in the previous steps, you can now proceed to bootstrap the platform before launching it.

To successfully execute several commands in the command line or in the terminal, you may need root-level access rights.

To obtain admin rights, run the following command(s):

```
$ sudo su -
```

Alternatively:

- Grant admin rights to a specific user, who can then log in with their password to perform admin tasks:

```
$ su - {user_name}
```

Or:

- Prefix `sudo` to the command you want to run:

```
$ sudo {command}
```

Load the Supervisor configuration

Launch Supervisor to load the configuration to start the tasks and processes other platform components depend on:

```
$ systemctl start supervisor
```

To check if the *supervisord* daemon is running, run the following command(s):

```
$ systemctl status supervisor
```

To check the statuses of the tasks managed by Supervisor, run the following command(s):

```
$ supervisorctl status
```

By default, Supervisor configuration files are stored in these locations:

- **Supervisor configuration file** (<http://supervisord.org/configuration.html#configuration-file>):
/etc/supervisor/supervisord.conf

- Configuration files of all Supervisor-managed applications: `/etc/supervisor/`
Make sure this location corresponds to the path defined in the `[include]` section of the `supervisord.conf` file:

```
[include]
# Default path for Ubuntu OS
files = etc/supervisor/*.ini
```

Typically, you can reach Supervisor by making a call to `http://localhost:9001` or to `http://127.0.0.1:9001`.

cURL call example:

```
$ curl http://localhost:9001
```

To make sure the platform and Supervisor can communicate, you may need to check and, if necessary, edit the `supervisord.conf` file.

Verify that the `supervisord.conf` file includes the following sections enabling the `supervisord` daemon to respond to requests from the platform:

- `[inet_http_server]` **section** (<http://supervisord.org/configuration.html#inet-http-server-section-settings>) to define the Supervisor host.
To enable `inet_http_server`, `supervisord.conf` should contain the following properties under `[inet_http_server]`:

```
[inet_http_server]
port=127.0.0.1:9001
```

- `[rpcinterface:supervisor]` **section** (<http://supervisord.org/configuration.html#rpcinterface-x-section-settings>) to allow communication between the Supervisor XML-RPC interface and the platform API to retrieve status information.
To enable `rpcinterface`, `supervisord.conf` should contain the following properties under `[rpcinterface:supervisor]`:

```
[rpcinterface:supervisor]
supervisor.rpcinterface_factory = supervisor.rpcinterface:make_main_rpcinterface
```

Example Supervisor configuration file:


```
[supervisord]
logfile=/var/log/supervisor/supervisord.log
logfile_maxbytes=5MB
logfile_backups=0
loglevel=info
pidfile=/var/run/supervisord.pid
nodaemon=false
minfds=40000
minprocs=200

[unix_http_server]
file=/var/run/supervisor/supervisor.sock

[inet_http_server]
port=127.0.0.1:9001

[rpcinterface:supervisor]
supervisor.rpcinterface_factory = supervisor.rpcinterface:make_main_rpcinterface

[supervisorctl]
serverurl=unix:///var/run/supervisor/supervisor.sock

[include]
# Default path for Ubuntu OS
files = etc/supervisor/*.ini
```

To make sure the platform and Supervisor can communicate, you may need to check and, if necessary, edit the `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` file.

- Verify that the `/opt/eclecticiq/etc/eclecticiq/platform_settings.py` file includes the following properties:

```
# Supervisor service address
# Comma separated list of 'host:port'
SUPERVISORD_HOSTS = '127.0.0.1:9001'

# List of services Systemd monitors
SYSTEMD_SERVICES = [
    'elasticsearch',
    'logstash',
    'postfix',
    'neo4j',
    'postgresql-9.5',
    'redis',
]
```



Warning: When you edit or update the Supervisor configuration, you need to run `$ supervisorctl reload` to update it and to reload the up-to-date configuration into the platform.

Examples of `supervisorctl` actions:

Run this...	...to do this
<code>\$ supervisorctl start all</code>	Start all the processes defined in the Supervisor configuration
<code>\$ supervisorctl start <process_name></code>	Start the specified process defined in the Supervisor configuration

Run this...	...to do this
\$ supervisorctl start <process_name> <process_name>	Start the specified processes defined in the Supervisor configuration
\$ supervisorctl stop all	Stop all the processes defined in the Supervisor configuration file
\$ supervisorctl stop <process_name>	Stop the specified process
\$ supervisorctl stop <process_name> <process_name>	Stop the specified processes
\$ supervisorctl status	Retrieve the statuses of the processes managed by Supervisor. For further information, see the official ocumentation on the return state values (http://supervisord.org/subprocess.html#process-states)
\$ supervisorctl status <process_name>	Retrieve the status of the specified process
\$ supervisorctl status <process_name> <process_name>	Retrieve the status of the specified processes
\$ supervisorctl reload	Reload the Supervisor configuration and restart all tasks and processes. If you modify or update the Supervisor configuration file, you need run this command to reload the latest Supervisor configuration file
\$ supervisorctl reload all	Reload all the processes managed by Supervisor. This command works just like \$ supervisorctl reload
\$ supervisorctl update	Restart the applications whose configuration has changed. This command affects existing configurations that were modified. If new application configurations become available, they start automatically only after restarting Supervisor or after rebooting the system. Typically, you run <code>reread</code> and then <code>update</code> .
\$ supervisorctl tail <log_file_name>	Retrieve the most recent lines of the specified log file. To follow the log file as it updates with new information and new lines, run \$ <code>supervisorctl tail -f <log_file_name></code>
\$ supervisorctl status grep "<search_string>"	Retrieve the status of all processes whose name contains the specified search string.
\$ supervisorctl reread	Update the changed configurations and reload them. It does not automatically (re)start any applications. Typically, you run <code>reread</code> and then <code>update</code> .

For further information on `supervisord` and `supervisorctl`, see the **official documentation** (<http://supervisord.org/running.html>).

Set up the database

If you are installing the platform for the first time, or if it is a fresh install, there is no database yet. To set up the database, you first need to create it, and then you load the default fixtures for the platform.

To create the database schema, run the following command(s):

```
$ /opt/eclecticiq/migrations/create-db.sh
```

To generate the default platform fixtures, run the following command(s):

```
$ /opt/eclecticiq/migrations/load-fixtures.sh
```

Bootstrap Elasticsearch

To make sure you are applying the latest Elasticsearch schema, fully reindex Elasticsearch.

- Make sure that ingestion, indexing, and core platform processes are *not running*. If they are running, stop them:

```
$ sudo supervisorctl stop intel-ingestion:* intel-search-indexer platform-api
```

- Set `elasticsearch` as the designated superuser to execute the reindexing command with:

```
$ sudo -u elasticsearch -i
```

- Configure the platform settings environment variable by prepending it to the `manage synchronize_elasticsearch` command.
- Run `manage synchronize_elasticsearch` to index or to fully reindex all Elasticsearch indexes. The command runs in the background. In case of an SSH disconnection, the process should keep running normally.

```
$ export INTELWORKS_PLATFORM_SETTINGS=/opt/eclecticiq/etc/eclecticiq/platform_settings.py  
$ /opt/eclecticiq/platform/api/bin/manage synchronize_elasticsearch > ./reindexing.log 2>&1 &
```

- Reindexing messages, if any, are logged to *reindexing.log*. An empty file means that the process completed successfully.

Bootstrap Neo4j

To bootstrap Neo4j, do the following::

- Create the graph schema.
- Apply the graph database migrations.

Prerequisites

Before proceeding to bootstrap Neo4j, verify that the following conditions are met:

- Make sure the Neo4j settings in the *platform_settings.py* configuration file are correct, based on your system environment.
Typical/Default values are:

```
NEO4J_URL: 'http://localhost:7474'  
NEO4J_BATCHING_URL = 'http://127.0.0.1:4008'  
NEO4J_DEBUG: False
```

neo4j-batching	Neo4j graph database batch processing application. It lives on the same server hosting the Neo4j database. It prepares data for ingestion into the Neo4j database
----------------	---

Neo4j should be up and running:

- Check if Neo4j is running by making a cURL call to the URL defined in `NEO4J_URL` in the *platform_settings.py* configuration file.
By default, it is `http://localhost:7474`.
- If Neo4j is not running, restart the service by running the following command(s):

```
$ systemctl start neo4j
```

- To check if Neo4j is running, run the following command(s):

```
$ systemctl status neo4j
```

Create the graph schema

- Before creating the graph schema, stop the *graph-ingestion* and any running *intel-ingestion* tasks:

```
$ supervisorctl stop graph-ingestion intel-ingestion:*
```

- Then, run the following command to create the graph schema and to apply the necessary graph database migrations to the Neo4j database:

```
$ export INTELWORKS_PLATFORM_SETTINGS=/opt/eclecticiq/etc/eclecticiq/platform_settings.py  
$ /opt/eclecticiq/platform/api/bin/manage do_graph_migrations
```

Load the dashboard

To populate the platform dashboard, you need to fetch data from Elasticsearch and Kibana. **elasticdump** (<https://github.com/taskrabbit/elasticsearch-dump>) helps you do that.

To load the platform dashboard, run the following command(s):

```
# Run elasticsearch to load the platform dashboard
$ elasticsearch --input=/opt/eclecticiq/etc/kibana-dashboards.json --output=http://localhost:9200/-kibana
```

Run a final check

As a last step before launching the platform, it is good practice to check the following points:

- Core processes and services
- Search, indexing and graph
- Availability
- To check if a core service is enabled to start at system bootup run the following command(s):

```
$ systemctl is-enabled <service_name>
```

- To check if a core service is running run the following command(s):

```
$ systemctl status <service_name>
```

- To start a core service run the following command(s):

```
$ systemctl start <service_name>
```

Check core processes and services

Nginx

- Verify that Nginx is up and running by checking the web server status:

```
$ systemctl status nginx
```

Supervisor

To check if the *supervisord* daemon is running, run the following command(s):

```
$ systemctl status supervisor
```

PostgreSQL

To check if PostgreSQL is running, run the following command(s):

```
$ systemctl status postgresql
```

or:

```
$ systemctl list-units | grep -i postgre
```

Check search indexing and graph

Elasticsearch

To check if Elasticsearch is running, run the following command(s):

```
$ systemctl status elasticsearch
```

Neo4j

Check the main connectivity properties to verify that Neo4j can communicate with other components.

Check neo4j-server.properties

- Open the *neo4j-server.properties* configuration file:

```
$ nano /etc/neo4j/neo4j-server.properties
```

- Edit the default property values, so that they reflect your system configuration:

```
org.neo4j.server.webserver.address=0.0.0.0
org.neo4j.server.webserver.port=7474
org.neo4j.server.database.location=/media/neo4j/graph.db
org.neo4j.server.webserver.https.enabled=false
dbms.security.auth_enabled=false
```

graph.db is the default name of the graph database file.

Make sure the directory where it is stored exists before setting it in this configuration file, and that Neo4j can read and write data to it.

Inspect the amount of allocated memory: Neo4j purrs https://www.youtube.com/watch?v=drq_ww7ytzw when you give it plenty of memory to work with.

- Depending on your system resources, allocate at least 4 GB to Neo4j.
- Never set it below 1 GB.

To set the minimum and maximum memory values for Neo4j, do the following:

Check neo4j-wrapper.conf

- Open the *neo4j-wrapper.conf* configuration file:

```
$ nano /etc/neo4j/neo4j-wrapper.conf
```

- Make sure that the following properties are set to the corresponding values:

```
wrapper.java.initmemory=1024
wrapper.java.maxmemory=4096
```

- The recommended value for `wrapper.java.initmemory` is at least 4096 MB or more, if possible. The minimum value should never be lower than 1024 MB.
- The recommended value for `wrapper.java.maxmemory` is at least 4096 MB or more, if possible. This value should never be lower than `wrapper.java.initmemory`.

Check neo4j.properties

Set the size of the page cache to 5G:

- Open the `neo4j.properties` property file:

```
$ nano /etc/neo4j/neo4j.properties
```

- Make sure the `dbms.pagecache.memory` property is set to 5G:

```
dbms.pagecache.memory=5G
```

If this parameter is not configured, Neo4j defaults to using 50% of the available system RAM for page caching.

Check availability

Make sure Elasticsearch and Neo4j are available by making cURL calls to the corresponding endpoints:

```
# Check Elasticsearch availability
$ curl localhost:9200

# Check Neo4j availability
$ curl localhost:7474
```

If Neo4j is not running, restart the service by running the following command(s):

```
$ systemctl start neo4j
```

Reload Supervisor configurations

- After completing the installation, manually restart Supervisor by first stopping it, and then by starting it. In this way, it can pick up any changed configurations before starting all managed applications:

```
$ systemctl stop supervisor

$ systemctl start supervisor
```

To check the statuses of the tasks managed by Supervisor, run the following command(s):

```
$ supervisorctl status
```

The response should return `RUNNING` for all relevant tasks to confirm that all Supervisor tasks are being executed normally.

The following example serves as a guideline:

```
graph-ingestion          RUNNING    pid 3443,  uptime 1:10:56
intel-ingestion:0        RUNNING    pid 3323,  uptime 1:11:19
intel-ingestion:1        RUNNING    pid 3336,  uptime 1:11:14
intel-ingestion:2        RUNNING    pid 3363,  uptime 1:11:09
intel-ingestion:3        RUNNING    pid 3372,  uptime 1:11:04
kibana                   RUNNING    pid 13837, uptime 9 days, 0:38:17
neo4j-batching           RUNNING    pid 3590,  uptime 1:10:45
opentaxii                RUNNING    pid 3662,  uptime 1:10:39
platform-api             RUNNING    pid 2999,  uptime 1:12:47
search-ingestion         RUNNING    pid 3513,  uptime 1:10:49
task:beat                RUNNING    pid 3102,  uptime 1:12:39
task:enrichers           RUNNING    pid 3110,  uptime 1:12:26
task:integrations        RUNNING    pid 3135,  uptime 1:12:13
task:providers           RUNNING    pid 3204,  uptime 1:12:01
task:reindexing          RUNNING    pid 3223,  uptime 1:11:48
task:utilities           RUNNING    pid 3242,  uptime 1:11:35
```



Warning: When you edit or update the Supervisor configuration, you need to run `$ supervisorctl reload` to update it and to reload the up-to-date configuration into the platform.

Access the platform

To access the platform, do the following:

- Launch a web browser (recommended: Google Chrome).
- Go to the configured platform address, for example: `https://platform.host`
- On the login page, enter the appropriate credentials.



Warning: The browser may display an untrusted connection warning: add it as an exception, and then proceed to the platform.

Backup guidelines

Back up platform data to restore it when you upgrade or reinstall the platform, and as a disaster recovery mitigation strategy.

As a best practice, we recommend you implement a backup strategy for platform data. Backups come in handy in situations like:

- Platform upgrade to a newer release
- Platform reinstallation
- Disaster recovery.

Before starting a platform backup, verify the following points:

- No users are signed in to the platform
- No read/write activity is in progress
- The platform is not running.

The core data you should include in a platform backup are:

- Configuration files
- Databases.

The exact steps to back up platform data may vary, depending on your specific environment hardware, software, configuration, and setup. Therefore, consider the following as a set of generic guidelines on backing up platform data.

To successfully execute several commands in the command line or in the terminal, you may need root-level access rights.

To obtain admin rights, run the following command(s):

```
$ sudo su -
```

Alternatively:

- Grant admin rights to a specific user, who can then log in with their password to perform admin tasks:

```
$ su - {user_name}
```

Or:

- Prefix `sudo` to the command you want to run:

```
$ sudo {command}
```

Platform configuration

Back up the platform configuration files to restore the platform setup at a later time.

To get a list with the platform configuration files, run the following command(s):

```
# Returns only core config files
$ find /opt/eclecticiq/etc/eclecticiq/ -type f

# Returns platform and third-party components config files
$ find /opt/eclecticiq/etc/ -type f

# Returns third-party config files to use for reference as boilerplates
$ find /opt/eclecticiq/etc-extras/ -type f
```

The response returns the configuration files to include in the backup:

```
# Core platform settings
/opt/eclecticiq/etc/eclecticiq/opentaxii.yml
/opt/eclecticiq/etc/eclecticiq/platform_settings.py
/opt/eclecticiq/etc/eclecticiq/proxy_url

# Dashboard settings
/opt/eclecticiq/etc/kibana-dashboard.json

# Logstash log aggregation settings
/opt/eclecticiq/etc/logstash/conf.d/filters.conf
/opt/eclecticiq/etc/logstash/conf.d/input.conf
/opt/eclecticiq/etc/logstash/conf.d/neo4j-batching.conf
/opt/eclecticiq/etc/logstash/conf.d/opentaxii.conf
/opt/eclecticiq/etc/logstash/conf.d/output.conf
/opt/eclecticiq/etc/logstash/conf.d/platform-api.conf
/opt/eclecticiq/etc/logstash/conf.d/platform-ui.conf

# Logstash heap size
/opt/eclecticiq/etc/default/logstash # Ubuntu
/opt/eclecticiq/etc/sysconfig/logstash # CentOS, RHEL

# Web server, proxy and port settings
/opt/eclecticiq/etc/nginx/conf.d/platform.conf

# Platform components ini file settings: ingestion, search, graph, taxii, api, tasks
/opt/eclecticiq/etc/supervisord.d/graph-ingestion.ini
/opt/eclecticiq/etc/supervisord.d/intel-ingestion.ini
/opt/eclecticiq/etc/supervisord.d/neo4j-batching.ini
/opt/eclecticiq/etc/supervisord.d/opentaxii.ini
/opt/eclecticiq/etc/supervisord.d/platform-api.ini
/opt/eclecticiq/etc/supervisord.d/search-ingestion.ini
/opt/eclecticiq/etc/supervisord.d/task-workers.ini

# Kibana config file
/opt/eclecticiq/etc-extras/kibana.yml

# Redis message broker config file
/opt/eclecticiq/etc-extras/redis.conf

# Elasticsearch config files
/opt/eclecticiq/etc-extras/elasticsearch/elasticsearch.yml
/opt/eclecticiq/etc-extras/elasticsearch/logging.yml

# Neo4j config files
/opt/eclecticiq/etc-extras/neo4j/neo4j.properties
/opt/eclecticiq/etc-extras/neo4j/neo4j-server.properties
/opt/eclecticiq/etc-extras/neo4j/neo4j-wrapper.conf

# Nginx web server config file
/opt/eclecticiq/etc-extras/nginx/nginx.conf

# Postfix email server config file
/opt/eclecticiq/etc-extras/postfix/main.cf

# Kibana ini file
/opt/eclecticiq/etc-extras/supervisord/kibana.ini
```

Platform databases

The EclecticIQ Platform uses the following databases:

Database	Description
PostgreSQL (http://www.postgresql.org/docs/manuals/)	Intel database. It stores all the information about entities, observables, relationships, taxonomy, and so on.
Elasticsearch (https://www.elastic.co/guide/index.html)	Indexing and search database. It stores document data and search queries as JSON.
Neo4j (http://neo4j.com/docs/)	Graph database. It stores node, edge, and property information to build and represent data relationships.

It is best to include all databases in your backup strategy. If for any reason it is not possible, make sure that at least the PostgreSQL database is backed up on a regular basis.

Backup guidelines

Back up the PostgreSQL database

You can back up a PostgreSQL database in several ways:

- SQL dump (recommended)
- File system level backup
- Continuous archiving

SQL dump

The quickest way to backup a PostgreSQL database is to perform an **SQL dump** (<http://www.postgresql.org/docs/current/static/backup-dump.html>). To generate an SQL dump of the database, run the **pg_dump** (<http://www.postgresql.org/docs/current/static/app-pgdump.html>) or the **pg_dumpall** (<http://www.postgresql.org/docs/current/static/app-pg-dumpall.html>) command.

To create an *.sql* dump file, run the following command(s):

```
$ sudo -u postgres /usr/pgsql-9.5/bin/pg_dumpall > /output/db_dump.sql
```

To restore a backed up database from an *.sql* dump file, run the following command(s):

```
$ psql -U postgres < ./db_dump.sql
```

File system level backup

File system level backup (<http://www.postgresql.org/docs/current/static/backup-file.html>) creates backup copies of the PostgreSQL file used to store the database data corpus. The files to back up are stored in the **database cluster** (<http://www.postgresql.org/docs/current/static/creating-cluster.html>) specified with the **initdb** (<http://www.postgresql.org/docs/current/static/app-initdb.html>) command during database storage location initialization.

This approach requires database downtime.

To back up a database by archiving the files PostgreSQL uses to store data in the database, run the following command(s):

- Go to the PostgreSQL data directory, typically `../pgsql/<version>/data/` (CentOS and RHEL), or `../postgresql/<version>/main/` (Ubuntu):

```
$ cd /etc/postgresql/9.5/main/
```

- Create a `.tar` archive containing the entire content of the `data` directory:

```
$ tar -cvf db_backup.tar *
```

To restore a database by copying the files PostgreSQL uses to store data in the database, run the following command(s):

- Copy the `.tar` archive you just created to the target environment.
- In the target environment, delete any content in the `data` directory:

```
$ rm -rf /etc/postgresql/9.5/main/*
```

- Go to the PostgreSQL data directory where you want to restore the database data, typically `../pgsql/<version>/data/` (CentOS and RHEL), or `../postgresql/<version>/main/` (Ubuntu):

```
$ cd /etc/postgresql/9.5/main/
```

- Decompress the `.tar` archive:

```
$ tar -xvf db_backup.tar
```

Continuous archiving

Continuous archiving (<http://www.postgresql.org/docs/current/static/continuous-archiving.html>) allows you to back up and restore a snapshot of the database in the state it was at a given point in time. It combines file system level backup with write-ahead logging (WAL).

These are the main steps you need to carry out to set up this backup strategy:

- Configure the **write-ahead log** (<http://www.postgresql.org/docs/current/static/wal-configuration.html>) behavior. Usually, a section in the `postgresql.conf` file contains the WAL parameters you need to define. For example you would typically set **wal_level** (<http://www.postgresql.org/docs/current/static/runtime-config-wal.html#guc-wal-level>) to **archive**, **archive_mode** (<http://www.postgresql.org/docs/current/static/runtime-config-wal.html#guc-archive-mode>) to **on**, and you may wish to set an **archive_command** (<http://www.postgresql.org/docs/current/static/runtime-config-wal.html#guc-archive-command>), as well as an **archive_timeout** (<http://www.postgresql.org/docs/current/static/runtime-config-wal.html#guc-archive-timeout>).

- Perform a **base backup** (<http://www.postgresql.org/docs/current/static/continuous-archiving.html#backup-base-backup>) by running **pg_basebackup** (<http://www.postgresql.org/docs/current/static/app-pgbasebackup.html>).
- As an alternative, you can manage backups with **pgbarman** (<http://www.pgbarman.org/>), an open source tool you can **download here** (<https://sourceforge.net/projects/pgbarman/>).

Back up the Elasticsearch database

The Elasticsearch official documentation includes sections with explanations of some **key concepts** (https://www.elastic.co/guide/en/elasticsearch/reference/current/_basic_concepts.html), as well as step-by-step tutorials on the following topics:

- **Back up an Elasticsearch database** (<https://www.elastic.co/guide/en/elasticsearch/reference/current/backup.html>)
- **Back up an Elasticsearch cluster** (<https://www.elastic.co/guide/en/elasticsearch/guide/current/backing-up-your-cluster.html>)
- Use the **snapshot API** (<https://www.elastic.co/guide/en/elasticsearch/reference/current/modules-snapshots.html>) to create snapshots of an index or a whole cluster.

Back up the Neo4j database

The Neo4j official documentation includes sections with explanations of **backup** (<http://neo4j.com/docs/stable/operations-backup.html>) concepts and procedures:

- **Configure** (<http://neo4j.com/docs/stable/backup-introduction.html>) Neo4j to enable backups
- **Back up** (<http://neo4j.com/docs/stable/backup-performing.html>) the Neo4j database
- Use **neo4j-backup**, the **Neo4j backup tool** (<http://neo4j.com/docs/stable/re04.html>) (shipped with the Neo4j Enterprise edition only).

Neo4j Community edition does not include a backup tool.

The following examples provide simple suggestions to manually start a backup operation.

- Before starting backing up data, stop Neo4j.
Run the following command(s):

```
$ systemctl stop neo4j
```

- Make sure Neo4j has stopped.
Run the following command(s):

```
$ systemctl status neo4j
```

- Once Neo4j is stopped, browse to the Neo4j data directory and compress the *graph.db* directory.
Run the following command(s):

```
$ cd <neo4j_data_dir>
$ tar -cfvz graph.db.tar.gz graph.db/
```

- Copy the compressed file to a different directory.

The Neo4j data location is defined in the */etc/neo4j/neo4j-server.properties* **configuration file** (<http://neo4j.com/docs/stable/server-configuration.html>) **as follows:**

```
org.neo4j.server.database.location=/media/neo4j/graph.db
```

Data recovery

To restore backed up data, follow the standard recommendations and procedures for PostgreSQL, Elasticsearch, and Neo4j:

- **Back up and restore PostgreSQL data** (<https://www.postgresql.org/docs/current/static/backup.html>)
- **Restore a PostgreSQL data dump** (<https://www.postgresql.org/docs/current/static/backup-dump.html#backup-dump-restore>)
- **Restore PostgreSQL data using a continuous archive backup** (<https://www.postgresql.org/docs/current/static/continuous-archiving.html#backup-pitr-recovery>)
- **Restore PostgreSQL using pg_restore** (<http://postgresguide.com/utilities/backup-restore.html>)
- **Restore Elasticsearch data with snapshot and restore** (<https://www.elastic.co/guide/en/elasticsearch/reference/current/modules-snapshots.html>)
- **Restore a Neo4j database backup** (<https://neo4j.com/docs/operations-manual/current/backup/#backup-restoring>)

Check for failed packages

During a data restore operation, you may wish to check if the scenario that created the need for a data backup restore also caused some packages to be partially or incorrectly ingested into the platform.

The blobs associated with problematic packages are not marked a successful. To retrieve a list with these packages, execute the following SQL query against the PostgreSQL database:

```
$ SELECT id, processing_status FROM blob WHERE processing_status NOT IN ('success', 'pending');
```

The query returns all packages whose status is not `success` or `pending`, as well as additional information on the packages, when available. Examine the response to evaluate whether you want to try reingesting the packages again.