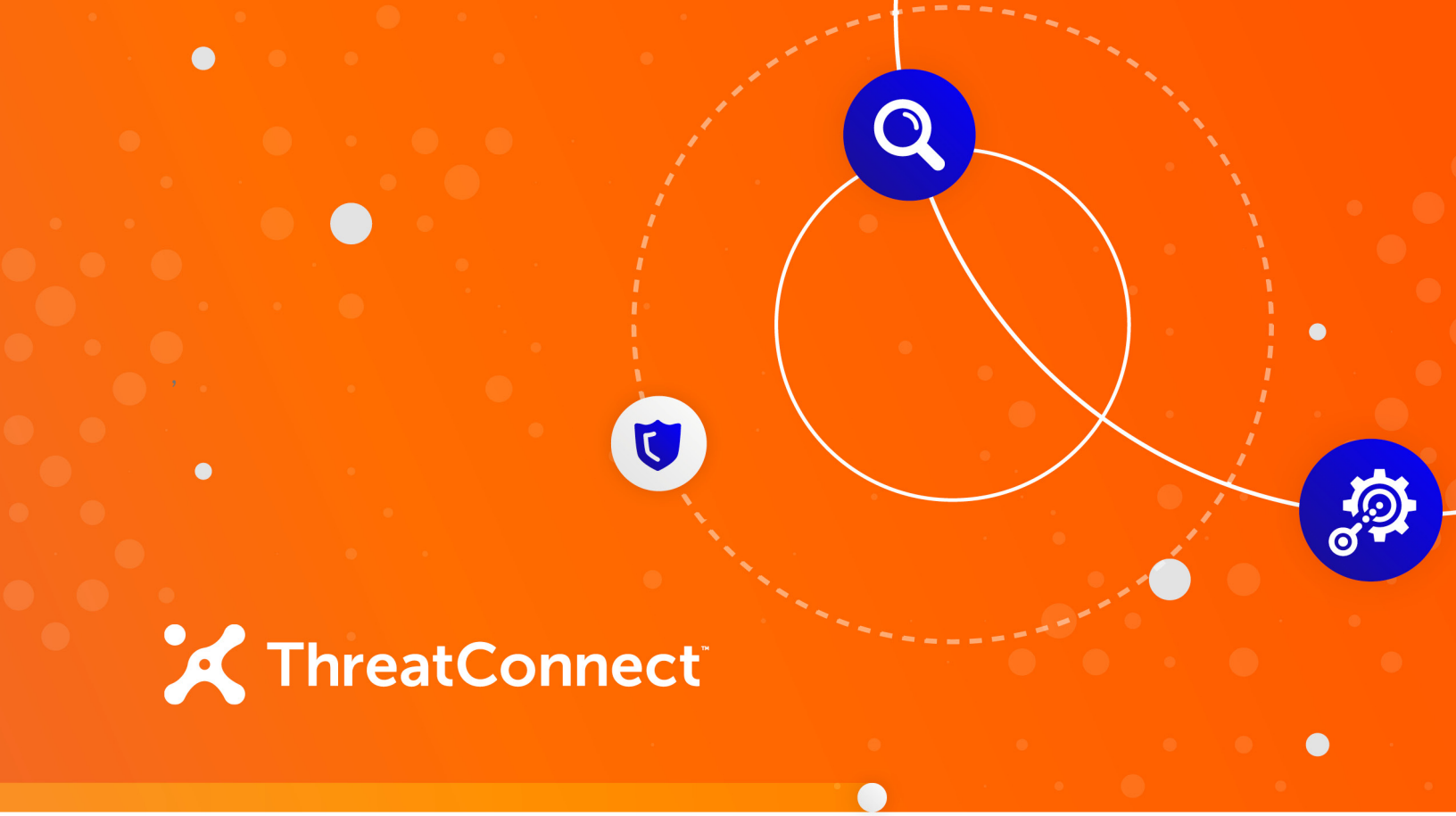

ThreatConnect™

Linux® Operating System

Installation Guide

SOFTWARE VERSION 6.0

MARCH 27, 2020

10015-08 EN Rev. C

ThreatConnect, Inc.

3865 Wilson Blvd., Suite 550, Arlington, VA 22203

P: 1.800.965.2708 | F: 1.703.229.4489

www.ThreatConnect.com

Table of Contents

SYSTEM REQUIREMENTS

To install an On-Premise instance of ThreatConnect®, the requirements in the following sections must be met.

Hardware

ThreatConnect requires a server, virtual or physical, that meets the specifications listed in Tables 1-5.

NOTE: Multi-server installations are for advanced users only, who should consult with ThreatConnect as to the correct sizing that will meet their needs. See the System Requirements Guide for additional information.

CAUTION: This document should be viewed in Google Chrome™ and not in Mac® OS Preview.

Table 1

	Playbooks	Memory Min (GB)[1,2]	Min CPU Cores / vCPUs (2GHz)[2]	Estimated Storage (GB)[3,4]
Application Server	No	16	8	50
	Yes	48	8	150

[1]Allocated to TC; OS needs extra. Large single sources (a source with 2+ Million indicators) may require more memory for bulk processing.

[2]Plus cores and memory indicated by installed apps. (This memory is not allocated to TC in the configuration since apps run in their OS process and require their memory).

[3]High IOPS, ideally SSDs, are preferred.

[4]ThreatConnect must be installed on an ext4 or XFS partition when running Playbooks.

Table 2

	Indicators (Millions)	Memory Min (GB)[1]	Min CPU Cores / vCPUs (2GHz)	Estimated Storage (GB)[1]
Database Server	0-2	12	6	20
	2-5	16	8	40
	5-10	32	12	60

[1]Allocated to the database; OS needs extra.

Table 3

	Indicators (Millions)	Memory Min (GB)	Min CPU Cores / vCPUs (2GHz)	Estimated Storage (GB)
Elasticsearch® Server	0-2	12	6	20
	2-5	16	8	40
	5-10	32	12	60

Table 4

Document Storage	Equal to the desired capacity of documents stored

Table 5

	Memory Minimum (GB)	Memory Recommended (GB)
Swap Space	4	8

NOTE: As the number of users increases, or as the frequency or complexity of automated analysis increases, the need to increase system resources will likely occur.

Software

ThreatConnect and its supporting packages require the following software to run correctly:

- **Operating System:** Red Hat® Linux variant—either Red Hat Enterprise Linux (RHEL) or Community Operating System (CentOS) 6 or 7
- **Oracle® Java® Development Kit (JDK):** Access to a local installation of Oracle Java 11 or OpenJDK (JDK version 11).
- **Elasticsearch**: Elasticsearch Server 6.3.0
- **Python®**: Installation of Python 3.6.x only

  NOTE: This refers to CPython. No other type of Python is permitted.
- **Python SDK:** TCEX version 2.0+
- **Redis**: Installation of Redis 5.0
- **Database:**

  NOTE: Select and install one of the databases listed below. Be aware that, depending on the distribution of ThreatConnect to be installed, users might not have the option to use MySQL® as the database.
  - **MySQL**: Installation of MySQL 5.7.X Community or Enterprise Edition
  - **SAP HANA®**: Installation of SAP HANA 2.0 SPS 02
  - **PostgreSQL**: Installation of PostgreSQL v11

Database

ThreatConnect requires an available instance of the MySQL 5.7 database, SAP HANA 2.0 SPS 02 database, or PostgreSQL v11 database. For MySQL and PostgreSQL, a client connection requires permissions to create users, databases, and tables within this instance during the installation process. Also, while it is acceptable to run one instance of the database on the same server as ThreatConnect, clients are advised to instantiate another machine for the replicated database. It is recommended that these machines conform to the hardware specifications above.

SMTP Server

ThreatConnect requires an available Simple Mail Transfer Protocol (SMTP) server to send email alerts and to correspond with users. This server must be routable from the server running the platform, and if SMTP authorization is required, ThreatConnect will need access to a username and password to generate these emails.

Web Browsers

The ThreatConnect platform supports up to two versions behind the current stable release of the following Web browsers:

- Google Chrome
- Mozilla Firefox®

ELASTICSEARCH INSTALLATION

Hardware

The hardware listed in Table 6 is recommended for a server to run Elasticsearch and its supporting plugins.

Table 6

	Indicators (Millions)	Memory (GB)	Min CPU Cores / vCPUs (2GHz)	Estimated Storage (GB)
Elasticsearch Server	0-2	12	6	20
	2-5	16	8	40
	5-10	32	12	60

Software

Elasticsearch and its supporting plugins require the following software environment to run correctly:

- **Operating System**: Any OS supported by the vendor
- **Java® Development Kit (JDK)**: Access to a local installation of the JDK (version 1.8 or newer)
- **Elasticsearch**: Elasticsearch Server 6.3.0

Additional Requirements

Gather the following information in advance:

- The server's IP address
- The total amount of RAM installed on the server

Network Configuration

By default, Elasticsearch will need ports 9200 and 9300 open to allow communication from the network. If Elasticsearch is to be installed on the same server as the ThreatConnect application, these ports do not need to be open.

Downloading and installing Elasticsearch

NOTE: The following installation instructions are for RHEL/CentOS Install only.

Download Elasticsearch from the following URL:

https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-6.3.0.rpm

Install the application by running:

```
rpm -Uvh elasticsearch-6.3.0.rpm
```

Verification

Refer to Table 7 to verify all Elasticsearch directories.

Table 7

Type	Directory
Base Directory	/usr/share/elasticsearch/
Logs	/var/log/elasticsearch/
Settings	/etc/elasticsearch/

If using non-standard installation directories, edit these values and refer to the upstream Elasticsearch documentation here:
https://www.elastic.co/guide/en/elasticsearch/reference/current/settings.html

Configuration

When allocating memory to Elasticsearch, the application should receive half of the total server memory. The other half of the server memory should be left intact for Lucene™ to use as storage. Lucene uses the underlying OS to cache data structures in memory, and its performance relies heavily on its ability to interact with the OS. If it does not have available memory to use, the full-text search will suffer performance impacts. The standard is to give half of the available memory to the Elasticsearch heap and leave the other half free for Lucene.

In the configuration file **jvm.options(default location /etc/elasticsearch/jvm.options)**, adjust the RAM to be allocated to Elasticsearch. Table 8 is an example of memory storage allocation. Users may have different amounts (GB) of RAM on their systems.

NOTE: If Elasticsearch is placed on a server that has more than 32GB of RAM, it is advised that the jvm.options settings are not set above 31g. Java is built to handle about 31.99GB of RAM optimally. Any allocation over 31.99GB will result in performance degradation.

Table 8

Physical RAM on System (GB)	2	4	8
Min Heap Size	-Xms1g	-Xms2g	-Xms4g
Max Heap Size	-Xmx1g	-Xmx2g	-Xmx4g

Open the configuration file (default location **/etc/elasticsearch/elasticsearch.yml**), and make the changes listed below.

Change the cluster name to **elasticTsearch**:

```
cluster.name: elasticTsearch
```

Add the hostname/IP of the machine:

```
network.host: <host>
```

NOTE: If Elasticsearch is to run on the same server as the ThreatConnect application, this value does not need to be specified. The Elasticsearch service will bind to the localhost IP and prevent any remote access to the Elasticsearch service.

Uncomment bootstrap.memory_lock = true:

```
bootstrap.memory_lock = true
```

Uncomment action.destructive_requires_name: true:

```
action.destructive_requires_name: true
```

Save and close the file.

Plugin Installation and Starting Elasticsearch

Install the Elasticsearch plugins in the elasticsearch/bin directory.

Ingest attachment: Allows for indexing of documents.

```
cd /usr/share/elasticsearch/bin/

./elasticsearch-plugin install ingest-attachment
```

If the download fails, download the file below to the server:

https://artifacts.elastic.co/downloads/elasticsearch-plugins/ingest-attachment/ingest-attachment-6.3.0.zip

Run the following command:

```
./elasticsearch-plugin install file:/<path-to-zip-file>
```

Start the Elasticsearch server:

```
service elasticsearch start
```

Check the logs to make sure that it started successfully:

```
tailf /var/log/elasticsearch/elasticTconnect.log
```

Verify that Elasticsearch can be accessed via a Web browser; it listens on port 9200 by default:

```
http://<host>:9200
```

MYSQL INSTALLATION AND CONFIGURATION

It is required to have a database instance operational before installing ThreatConnect. This section will outline how to configure the MySQL database to prepare it for usage with ThreatConnect.

MySQL Download and Installation

To acquire the latest version of MySQL 5.7, download and install a file that has the distribution repository configured.

```
# wget http://repo.mysql.com/mysql57-community-release-el7-11.noarch.rpm
# rpm -ivh mysql57-community-release-el7-11.noarch.rpm
```

NOTE: This file and its location may change, based on the application provider. Please consult Customer Service if the link does not function.

With the repository now installed and active, utilize **yum** to install the MySQL server:

```
# yum install mysql-server -y
```

Once the installation is complete, enable on startup and start the MySQL service:

```
# systemctl enable mysqld
# systemctl start mysqld
```

On the first start, MySQL will run through its first run process. In doing so, it will create a random password for the root account as part of the updated security measures for MySQL 5.7. To obtain this password, retrieve it from the log file with this command:

```
# grep "temporary password" /var/log/mysqld.log
```

The output string will define the root password. Copy this output to a file for future use.

Once the temporary root password is obtained, run the secure installation process to harden MySQL further:

```
# mysql_secure_installation
```

The temporary root password will be set as expired and will require an update. The password complexity requires that the new password be at least four characters long, including one numeric character, one lowercase character, one uppercase character, and one special (non-alphanumeric) character.

Once the password has been changed, the secure install process will continue:

- Enter **N** on "Change the password for root?"
- Enter **Y** on "Remove anonymous users?"
- Enter **Y** on "Disallow root login remotely?"
- Enter **Y** on "Remove test database and access to it?"
- Enter **Y** on "Reload privilege tables now?"

The secure installation is now complete.

MySQL Configuration Options

ThreatConnect requires the following options to be configured for **[mysqld]** in the MySQL configuration file **my.cnf**:

```
sql_mode=NO_ZERO_IN_DATE,NO_ZERO_DATE,NO_ENGINE_SUBSTITUTION
lower_case_table_names=1
character_set_server=utf8mb4
collation_server=utf8mb4_unicode_ci
group_concat_max_len=1000000
transaction_isolation=READ-COMMITTED
innodb_flush_log_at_trx_commit=2
innodb_table_locks=0
innodb_autoinc_lock_mode=2
eq_range_index_dive_limit=200
innodb_large_prefix=on
innodb_file_format=barracuda
innodb_buffer_pool_size=1G
event_scheduler=1
innodb_lock_wait_timeout = 500
```

NOTE: Set** innodb_buffer_pool_size **to a value around 75% of the database memory recommended in Table 1 (e.g., innodb_buffer_pool_size=1G).

Comment out the existing **sql_mode** line with a **#**. Save the file and restart the mysql service to commit the changes made in **my.cnf**:

```
# systemctl restart mysqld
```

Port Configuration

To allow access to the MySQL database from other servers, the firewall must be set up to allow incoming and outgoing connections:

```
firewall-cmd --permanent --zone=public --add-service=mysql
firewall-cmd --reload
```

Alternative methods may be used for the port configuration, but port 3306 needs to be open to allow servers to connect and relay data to the MySQL database.

Creating the Database and Users

Once the MySQL database service is installed and running, log in to create a database and user for ThreatConnect. From the command line on the MySQL database server, log in as the root user:

```
# mysql -u root –p
```

When prompted, enter the root password chosen upon installation. Enter the command below to create a new database. In this example, the database name **threatconnect** was chosen.

```
mysql> CREATE DATABASE threatconnect CHARACTER SET utf8mb4 COLLATE
utf8mb4_unicode_ci;
```

NOTE: Any database name may be used, but it will be referenced later in the installation and configuration process.

Once the database is created, a separate non-root user should be created for security reasons. In this example, the username **tcuser** and password **Password1!** were chosen. The **@'%'** addition allows **tcuser** to authenticate from any host on the network. A hostname can be specified here to limit possible login origins (e.g., localhost). Enter the command below to create a new user:

```
mysql> CREATE USER 'tcuser'@'%' IDENTIFIED BY 'Password1!';
```

Grant permissions on the newly created user, allowing them to access the tables within this database. Enter the following commands to log in to MySQL and add privileges to the new user:

```
mysql> GRANT ALL PRIVILEGES ON threatconnect.* TO 'tcuser'@'%' IDENTIFIED BY
'Password1!';
mysql> FLUSH PRIVILEGES;
mysql> quit;
```

NOTE: The above example assumes that the database was named threatconnect ***and the user*** tcuser***. It also assumes that*** tcuser's ***access is not limited to a specific host.***

NOTE: Any username/password may be used, but it will be referenced later in the installation and configuration process.

Creating ThreatConnect Tables

Before starting on this step, download the ThreatConnect binary zip file and have it extracted. The **threatconnect/app/scripts/mysql** directory contains a **.sql** script to create the initial tables for ThreatConnect within the database just defined. Browse to the folder containing the threatconnect-<version>**.sql** file, run the command below and enter the root password again when prompted.

NOTE: This command requires the use of SUPER permissions to run the script. If the MySQL root account is not to be used to run the script, make sure that the user account running the script has the proper permissions. The SUPER permissions are only needed for the non-root user account in MySQL when running the script. The permissions can be removed once the script import has completed.

```
# mysql -u root -p threatconnect < threatconnect-6.<version>.sql
```

NOTE: This command specifies threatconnect ***as the name of the database. If a different name for the above database was chosen, use that instead.***

THREATCONNECT INSTALLATION AND CONFIGURATION

Preparing the ThreatConnect Server

Open File Limits for a Standalone ThreatConnect Server

For a server only running ThreatConnect, edit this file location:

```
/etc/security/limits.conf
```

Add this line at the bottom:

```
threatconnect - nofile 12500
tc-job - nofile 12500
redis - nofile 10000
```

For this file location:

```
/etc/sysctl.conf
```

Add this line at the bottom:

```
fs.file-max = 40000
```

NOTE: Some CentOS/Red Hat versions come with strict permissions for the number of open files allowed per user (https://access.redhat.com/solutions/61334). The steps outlined adjust the number as needed for ThreatConnect to function. The above commands are based on using an account named *threatconnect* ***to run the ThreatConnect service,*** *tc-job user*, ***to be used within ThreatConnect for jobs and an account named*** *redis* ***to run the Redis service. The*** *fs.file-max* ***parameter is the maximum number of files that can be open in the OS.***

Open File Limits for an All-in-One ThreatConnect Server

For a server that is running ThreatConnect, the database and Elasticsearch edit this file location:

```
/etc/security/limits.conf
```

Add this line at the bottom:

```
threatconnect - nofile 12500
tc-job - nofile 12500
redis - nofile 10000
```

For this file location:

```
/etc/sysctl.conf
```

Add this line at the bottom:

```
fs.file-max = 150000
```

NOTE: Some CentOS/Red Hat versions come with strict permissions for the number of open files allowed per user (https://access.redhat.com/solutions/61334). The steps outlined adjust the number as needed for ThreatConnect to function. The above commands are based on using an account named** threatconnect **to run the ThreatConnect service,** tc-job user **to be used within ThreatConnect for jobs and an account named** redis **to run the Redis service. The** fs.file-max **parameter is the maximum number of files that can be open in the OS. The** fs.file-max **parameter is larger than the standalone configuration to accommodate for the open files needed for Elasticsearch and MySQL to function.

JDK

The system needs access to one of the JDKs as outlined in the *Software* section. Also, the **JAVA_HOME** environment variable needs to be appropriately configured to point to that directory.

Users must execute the following commands to verify Java is accessible:

```
# which java
```

User Account Creation

It is suggested that the ThreatConnect service be run by a service account. The steps below will assist in the creation of the **threatconnect** user account, which is used by default in the ThreatConnect software. The user account that will run the ThreatConnect service can be adjusted to run as another account, if needed.

To create the **threatconnect** account:

```
adduser threatconnect
```

To capitalize on the Java changes, update the user account's .bashrc. First, switch to the new user account:

```
su - threatconnect
```

Access the user's .bashrc:

```
vi ~/.bashrc
```

Add this line to the file if using Oracle Java JDK:

```
export JAVA_HOME=/usr/java/latest
```

Add this line to the file if using OpenJDK:

```
export JAVA_HOME=/usr/lib/jvm/jre-11-openjdk-11.0.6.10-1.el7_7.x86_64

NOTE: the above location will vary depending on OS version and version of
OpenJDK Java 11 downloaded to the system.

Restart the .bashrc with the new changes:
source ~/.bashrc
```

Verify that the changes are available:

```
echo $JAVA_HOME
```

Ensure that the changes match what was added. If they do, log out of the threatconnect user and continue.

Port Forwarding

For security reasons, the ThreatConnect application runs as an unprivileged user, which does not require root or Administrator permissions. Many operating systems prevent unprivileged applications from binding on commonly used ports, such as ports 80, 443, and 25, which are widely used by Web applications, such as ThreatConnect.

As a result, it is essential to configure redirection so that incoming web and email traffic on ports 80, 443, and 25 are forwarded to ports used by ThreatConnect (8080, 8443, and 2500, respectively).

Add the following rules to the firewallD:

```
systemctl enable firewalld
systemctl start firewalld
firewall-cmd --permanent --zone=public --add-service=smtp
firewall-cmd --permanent --zone=public --add-service=http
firewall-cmd --permanent --zone=public --add-service=https
firewall-cmd --permanent --zone=public --add-forward-
port=port=25:proto=tcp:toport=2500
firewall-cmd --permanent --zone=public --add-forward-
port=port=80:proto=tcp:toport=8080
firewall-cmd --permanent --zone=public --add-forward-
port=port=443:proto=tcp:toport=8443
firewall-cmd --permanent --direct --add-rule ipv4 nat OUTPUT 0 -p tcp -o lo -
-dport 443 -j REDIRECT --to-ports 8443
firewall-cmd --reload
```

Or add the following rules to the iptables:

NOTE: These rules will not persist after an OS reboot. It is suggested that** firewall-cmd **be used for persistent rules.

```
iptables -t nat -A PREROUTING -p tcp -m tcp --dport 443 -j REDIRECT --to-
ports 8443
iptables -t nat -A PREROUTING -p tcp -m tcp --dport 80 -j REDIRECT --to-ports
8080
iptables -t nat -A PREROUTING -p tcp -m tcp --dport 25 -j REDIRECT --to-ports
2500
iptables -t nat -A OUTPUT -p tcp -o lo --dport 443 -j REDIRECT --to-ports
8443
```

Redis Server Installation

NOTE: Redis, as other databases used by ThreatConnect software (e.g., MySQL and Elasticsearch), is not supported by ThreatConnect. Therefore, its support is technically independent of the company's product.

NOTE: Redis is installed on the application server.

It is required to have developer packages installed to compile Redis from source:

```
yum install –y bison byacc cscope ctags cvs diffstat doxygen flex gcc gcc-c++
gcc-gfortran gettext git indent intltool libtool patch patchutils rcs redhat-
rpm-config rpm-build tcl
```

Download the source code:

```
wget http://download.redis.io/releases/redis-5.0.7.tar.gz
Decompress the downloaded archive:
Tar -xvzf redis-5.0.7.tar.gz
```

Navigate to the directory where the decompressed files reside:

```
cd redis-5.0.7.tar.gz
```

Below are the compiled dependencies needed for Redis:

```
cd deps/
make hiredis lua jemalloc linenoise
```

Begin the compile process to ensure there are no errors:

```
cd ..
make && make test
```

Compile all modules needed:

```
make install
```

Issue the command below as a privileged user, and answer the particular questions to complete the install:

```
cd utils/
./install_server.sh
```

Alternatively, it can be started via **redis-server** passing in the **redis.conf** file located in **/temp/redis-5.0.7/redis.conf**.

Setting up the Redis Server

This script will help set up a running Redis server:

```
[root@localhost redis-5.0.7]# utils/install_server.sh
```

Select the Redis port for this instance **[6379]:**

```
Selected default: 6379
```

Select the Redis config file name **[/etc/redis/6379.conf]**:

```
Selected default - /etc/redis/6379.conf
```

Select the Redis log file name **[/var/log/redis_6379.log.]**:

```
Selected default - /var/log/redis_6379.log
```

Select the data directory for the instance **[/var/lib/redis/6379]**:

```
Selected default - /var/lib/redis/6379
```

Select the Redis executable path **[/usr/local/bin/redis-server]**.
Selected config:

```
Port: 6379
Config file: /etc/redis/6379.conf
Log file: /var/log/redis_6379.log
Data dir: /var/lib/redis/6379
Executable: /usr/local/bin/redis-server
Cli Executable: /usr/local/bin/redis-cli
```

Is this okay? Press **Enter** to continue, or press **Ctrl-C** to abort.

```
Copied /tmp/6379.conf => /etc/init.d/redis_6379
```

Installing service...Successfully added to chkconfig! **Successfully added to** runlevels 345**!**
Installation successful!

Once the configuration is completed, edit the configuration file:

```
vi /etc/redis/6379.conf
```

Add these lines at the end of the file:

```
maxmemory 6gb
maxmemory-policy allkeys-lru
maxmemory-samples 5
```

Save and close the file. To better troubleshoot and log any issues, run the Redis service as another user instead of root. To do so, create a new user account:

```
adduser redis
```

With the new user account created, edit the Redis service to run as the new Redis user. Edit the service by:

```
vi /etc/init.d/redis_6379
```

Add this line after REDISPORT:

```
USER=redis
```

Comment out this line:

```
$EXEC $CONF
```

Create a new line after the line that was just commented and add this:

```
su - $USER -c "$EXEC $CONF"
```

Save and close the file.

Change ownership of the following file and directories to the redis user:

```
chown -R redis:redis /var/lib/redis/
chown redis:redis /var/log/redis_6379.log
chown -R redis:redis /etc/redis/
```

Restart the Redis to apply the new changes made:

```
/etc/init.d/redis_6379 restart
```

Python Installation

NOTE: The instructions below compile and install Python from source code. This is one of the multiple ways Python can be installed on the application server.

It is required to have developer packages installed to compile Python from source:

```
yum install –y zlib-devel bzip2-devel openssl-devel ncurses-devel sqlite-
devel readline-devel tk-devel gdbm-devel db4-devel libpcap-devel xz-devel
expat-devel python-setuptools
```

Download the source code:

```
wget https://www.python.org/ftp/python/3.6.8/Python-3.6.8.tar.xz
```

Decompress the downloaded archive:

```
tar -xf Python-3.6.8.tar.xz
```

Navigate to the directory where the decompressed files reside:

```
cd Python-3.6.8
```

Run the following command to configure Python:

```
./configure --prefix=/usr/local --enable-shared LDFLAGS="-Wl,-rpath /usr/local/lib"
```

Begin the compile process to ensure there are no errors:

```
make && make altinstall
```

Setup symbolic link:

```
ln -s /usr/local/bin/python3.6 /usr/local/bin/python
```

Python SDK: TcEx Installation

NOTE: The instructions below need to be run after Python has been installed on the application server.

NOTE: The instructions below are based on the Python installation method discussed in this guide. Adjust directories as needed if Python is installed in another location.

To ensure no permissions issues arise from the use of Python packages, use the following commands to update permissions:

```
chmod -R 755 /usr/local/lib/python3.6/site-packages
chmod -R 755 /usr/local/lib/python3.6/lib2to3
```

To install tcex using pip:

```
/usr/local/bin/pip3.6 install tcex
```

Getting Started

ThreatConnect clients can use this guide to configure and install their instance of ThreatConnect. This guide assumes a moderate level of systems-administration expertise and an operating environment that satisfies the requirements detailed above.

NOTE: All references to** threatconnect-<version>* **should not be taken literally; instead, replace the** *<version>* **string with the most recent version of the software (e.g., threatconnect-6.0).

Downloading the Installer

To download the required ThreatConnect files, users will need their Onehub® credentials from their ThreatConnect representative.

1. Access the ThreatConnect Workspace **Sign In** screen (Figure 1) by navigating to: https://ws.onehub.com/workspaces/571735/signin. Once there, enter the required credentials.

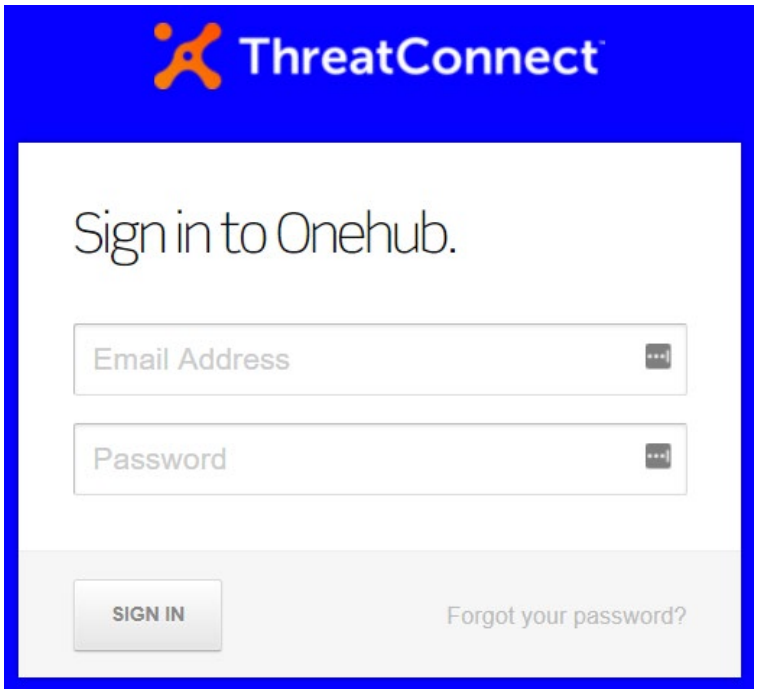

Figure 1

2. Click the **SIGN IN** button, and the **ThreatConnect Onehub Workspace** screen will appear (Figure 2).

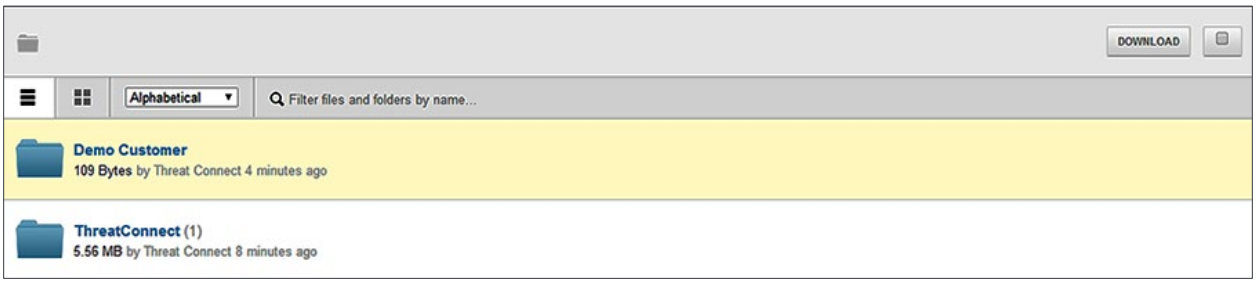

Figure 2

3. The screen displays two directories: a ThreatConnect folder and a private folder that is specific to each client's account.

The ThreatConnect Folder

The **ThreatConnect** folder contains all of the files required to install each version of ThreatConnect. The directory structure for this folder is as follows:

/ThreatConnect/

Release/

<version> - (Date)/

Documents/

threatconnect-<version>.zip

To download a file, click its corresponding checkbox (on the right side of the screen), and then click the **Download** button (Figure 3).

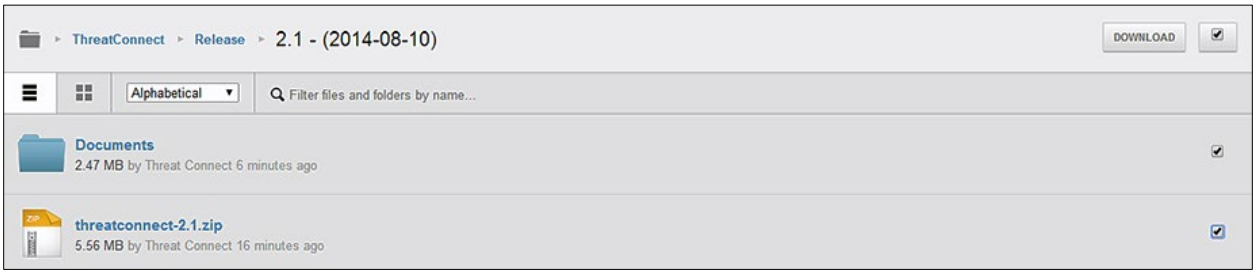

Figure 3

The Customer Folder

The **Customer** folder (in this example **Demo Customer**, Figure 4) is private between clients and their ThreatConnect representatives; other clients cannot access this folder or even know of its existence. From this folder, clients can download their license key. Additionally, the folder contains a sub-folder named **Upload**, which clients can use to safely and securely share files, such as screen captures or log files, which facilitates remote assistance from ThreatConnect representatives.

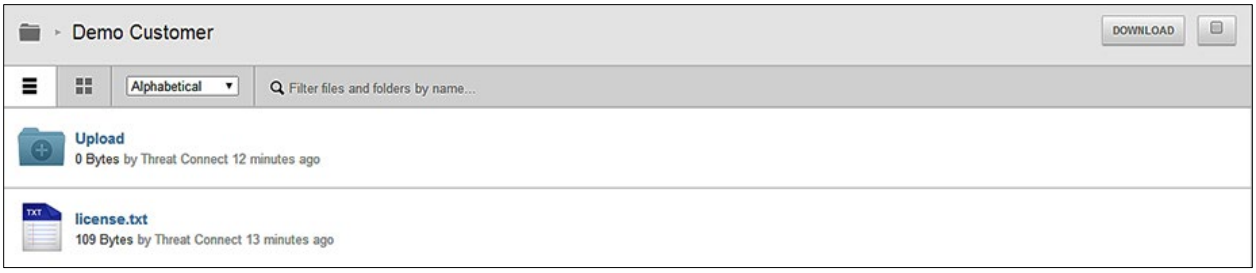

Figure 4

Opening the Installer

Unzipping the File

The ThreatConnect **.zip** file serves as an archive of all the necessary files and folders needed to install the platform.

1. Copy the **.zip** file to the desired directory on the ThreatConnect server. By default, this is **/opt**, which will result in an installation directory of **/opt/threatconnect**.
2. Unzip the file from the command-line interface with the following command:

```
unzip threatconnect-<version>.zip.
```

Installation Directory Structure

After extracting the archive, the new folder will contain the following directory structure:

threatconnect/

- **app/**
- **config/**
- **exchange/**

The **app** directory contains all of the binaries and supporting libraries needed to run ThreatConnect, and it also contains the scripts required to configure and administer a user's ThreatConnect instance.

The **config** directory contains all of the customized configuration information for a user's ThreatConnect instance. It is populated with keystore, XML, and properties files after the installer is run. The **app** and **config** directories are referenced throughout the rest of this document.

NOTE: Users patching their ThreatConnect instance to a newer version will need to replace the `app` directory with one from the latest version but will not need to update their `config` directory after it is established from the initial installation.

Folder Permissions Adjustment

Once the ThreatConnect binary has been unzipped and put in the location from where it will run, the permissions will need to be adjusted to allow the **threatconnect** user ownership of the files:

```
chown -R threatconnect:threatconnect /opt/threatconnect
```

NOTE: The above command is being used with the default installation path of /opt/threatconnect ***and the default user account of*** threatconnect***. Adjust the command above as needed.***

TC Exchange Setup

Creating "tc-job" User

To provide additional security, it is recommended that a separate user be created to run the TC Exchange™ jobs. It is also recommended that read and write groups be created to control the permission to these file:

```
useradd tc-job
echo "tc-job-pass123" | passwd tc-job --stdin
groupadd tc-job-read
usermod -a -G tc-job-read tc-job
chgrp -R tc-job-read /opt/threatconnect/exchange/programs
chmod -R 755 /opt/threatconnect/exchange/programs
groupadd tc-job-write
usermod -a -G tc-job-write tc-job
chgrp -R tc-job-write /opt/threatconnect/exchange/jobs
chmod -R 777 /opt/threatconnect/exchange/jobs
chmod +t /opt/threatconnect/exchange/jobs
```

Update permissions:

```
chown -R threatconnect:tc-job-read
/opt/threatconnect/exchange/programs/organization/
```

User Privilege Configuration

Add the following lines to the **pam** configuration above the top **auth** command:

```
/etc/pam.d/su
auth    sufficient pam_rootok.so
auth    [success=ignore default=1] pam_succeed_if.so user = tc-job
auth    sufficient pam_succeed_if.so use_uid user = threatconnect
```

Configuring Sudoers

Add the following to the sudoers configuration to allow the **threatconnect user** to run the jobs as the **tc-jobs user**.

```
/etc/sudoers.d/threatconnect
Defaults:threatconnect !requiretty
threatconnect ALL=(tc-job) NOPASSWD: ALL
```

Starting the Installer

Open a terminal window and browse to the **app** directory within the **ThreatConnect** folder. Run the **setup.sh** file as the ThreatConnect user defined for the application. The XML file can be corrupted if run as any other user:

```
# ./setup.sh
```

NOTE: Attempting to run the `start.sh` script before the respective `setup.sh` script will result in an error.

Installation Basics

The installer prompt allows for configuration of the database and SMTP servers used by ThreatConnect. Clients will be prompted to conduct an initial setup, and subsequent runs of the installer will present additional configuration options, as detailed in the following sections.

Time Zone

NOTE: Users MUST install the ThreatConnect instance in the UTC (Coordinated Universal Time) time zone for all servers, as the timestamps will be sent in UTC.

Use the commands below to change the time zone:

CentOS6 or RHEL 6:

```
mv /etc/localtime /etc/localtime_backup ln -s /usr/share/zoneinfo/UTC
/etc/localtime
```

CentOS7 or RHEL7:

```
timedatectl set-timezone UTC
```

For CentOS 6/7 and RHEL 6/7, once the change has been implemented, this can be validated via:

```
ls -l /etc/localtime
```

The expected output will look like:

```
lrwxrwxrwx 1 root 19 Nov 17 2018 /etc/localtime -> /usr/share/zoneinfo/UTC
```

Initial Setup

The initial setup runs the database, SMTP, and memory-configuration options simultaneously. It will reflect a user's input in the appropriate files within the **config** directory. After running the initial setup, additional advanced options will appear when running the setup script.

Selecting Appropriate Server Type

Select the appropriate server type to configure, which will construct the correct corresponding **threatconnect.xml** file to use on that server.

There are four options to choose from:

- Messaging server
- Playbooks/Job server
- Web/API server
- Full server

NOTE: The selection will always be the Full server option unless specified by the Deployment Engineer.

Configuring Database Properties

This option allows for the configuration of the database properties for ThreatConnect. To fulfill its role as an intelligence platform, ThreatConnect requires access to a database instance for storing Indicators, user data, and more. Clients will be prompted to enter the information in Table 9.

Table 9

Property	Description	Example Value
Database Name	Name of the database instance	threatconnect
Database Port	The port used by the database server	3306
Database Host	The hostname of the server hosting the database	localhost
Username	The username used for authentication with the database	tcuser
Password	The password used for authentication with the database	Password1!

The installer will offer the user the option of testing the database connection with the values as configured using Java Database Connectivity (JDBC). When prompted, save the database configuration changes if satisfied with the configuration results.

NOTE: If a valid database connection cannot be reached, then the installer will not proceed with the installation and configuration.

Configuring SMTP Properties

This option allows the user to set the SMTP server settings for ThreatConnect. The platform will use this server to send email alerts, communicate with users, and more. When setting the SMTP properties, users will be prompted for the information in Table 10.

Table 10

Property	Description	Example Value
Mail SMTP Host	The hostname of the SMTP server used by ThreatConnect	smtp.acme.net
Mail SMTP Port	The port used by the SMTP server	25
SSL Required	Whether the SMTP server requires encrypted SSL	true
SMTP User	The SMTP username, if authentication is required	tcsender
SMTP Password	The password used to authenticate the above username	smtppassword

NOTE: If user authentication is not needed for the SMTP settings, a value for the username and password still needs to be entered. These values can be removed from the threatconnect.xml file after the configuration is completed.

Configuring Memory Properties

This option allows the user to set the memory settings for the server. The ThreatConnect application will be generated using the parameters specified in Table 11. Once entered, the installer will confirm the settings.

Table 11

Property	Description	Example Value (GB)
Heap Size Limit	The memory limit for the heap	8
Initial Heap Size	The initial heap size	8

NOTE: Setting the heap size limit to less than 6 gigabytes (represented as 6G or 6000M) or higher than 31 gigabytes (represented as 31G or 31000M) may result in adverse performance. Use the same value for the initial heap size.

Configuring a Full Server

When configuring a **Full** server, the user will be prompted for the **Playbooks** server Worker size (e.g., 4). This value can be changed later in the UI if additional workers are needed.

Configuring the ThreatConnect License

This option allows the user to import a valid ThreatConnect license straight from the installer. Users will be asked if they wish to apply for a license at this point. If so, the absolute path of the ThreatConnect license file must be provided (e.g., /opt/threatconnect/license.xml).

STARTING THREATCONNECT

After completing the installation and configuration procedures, it is time to start ThreatConnect. The options in the previous sections allow a user to run ThreatConnect in a single session. This presents some limitations: the platform will need to be started manually after each reboot or a terminal window or Secure Shell (SSH) session may have to be left open. This section, thus, details the file configuration to run ThreatConnect as a service in Linux.

Open a terminal window and browse to the **app** directory within the ThreatConnect directory. Run the **start.sh** file as the ThreatConnect user defined for the application:

```
# ./start.sh
```

NOTE: It is suggested that the application be set up to run as a service. This limits any issues in configuration as well as allowing to have the application run in the background.

The Service Script

Within the ThreatConnect installation there is a script used for running ThreatConnect as an initialized service:

```
<threatconnect_home>/app/service/threatconnect.
```

This script must be copied into the **/etc/init.d** directory for it to be recognized as a system service. Note that users may need privileges to copy to this directory:

```
# cp <threatconnect_home>/app/service/threatconnect /etc/init.d
# chmod 755 /etc/init.d/threatconnect
```

Configuring Services

The service script requires proper permissions and paths to be set.

1. Specify the **BASEDIR** variable within the script to point to the path where the ThreatConnect installation exists. By default, this **is /opt/threatconnect**.
2. Specify the **USER** variable within the script to identify which user owns the files for the ThreatConnect application. It is advisable, for security reasons, that the root user not be employed. By default, the username is assumed to be **threatconnect**.

3. Optionally, un-comment the **iptables** lines below within the script file. Choosing to do so will redirect ports 443 and 80 to run on the specified ports. By default, they redirect to 8443 and 8080, although these may be modified, as needed, depending on how the ports were configured according to the Port Configuration section. The lines in the script are initially:

```
# iptables -t nat -A PREROUTING -p tcp -m tcp --dport 443 -j REDIRECT --
to-ports 8443
# iptables -t nat -A PREROUTING -p tcp -m tcp --dport 80 -j REDIRECT --to-
ports 8080
# iptables –t nat –A OUTPUT –p tcp –o lo –dport 443 –j REDIRECT –to-ports
8443
# iptables -t nat -A PREROUTING -p tcp -m tcp --dport 25 -j REDIRECT --to-
ports 2500
```

NOTE: If the** firewall-cmd **service is not allowed on the application server, it is suggested that the above rules be uncommented, so that network traffic is allowed through anytime the ThreatConnect service is online.

Starting ThreatConnect as a Service

Once the services have been configured, ThreatConnect as a service can be started. To do so, enter one of the following commands as the root user:

```
# service threatconnect start
# /etc/init.d/threatconnect start
# systemctl start threatconnect
```

To stop the service, enter any of the following:

```
# service threatconnect stop
# /etc/init.d/threatconnect stop
# systemctl stop threatconnect
```

To have ThreatConnect start on system startup, issue the following commands after the script is configured in the **/etc/init.d** directory:

```
# chkconfig --add threatconnect
# chkconfig threatconnect on
or for CentOS 7 / RHEL 7
systemctl enable threatconnect
```

THE FIRST LOGIN

Once the ThreatConnect instance is installed, configured, and running, log in to begin customization. Most of the administrative functions within ThreatConnect are outside the scope of this document and can be found in the *ThreatConnect Administration Guides*. But before doing so, complete the tasks in the following sections.

Entering Initial Login Credentials

To conduct the initial login, and to access a system account to begin customization, go to the ThreatConnect Instance in a Web browser, and log in with the username **admin** and the password **password1**. Change these credentials after the initial installation to enhance security.

Setting Master Key for Keychain

If the keychainEnabled property is set to **true**, a System Administrator will need to log out and log back into ThreatConnect to set the master key. When prompted, enter a Master Secret Key in the fields provided (Figure 5).

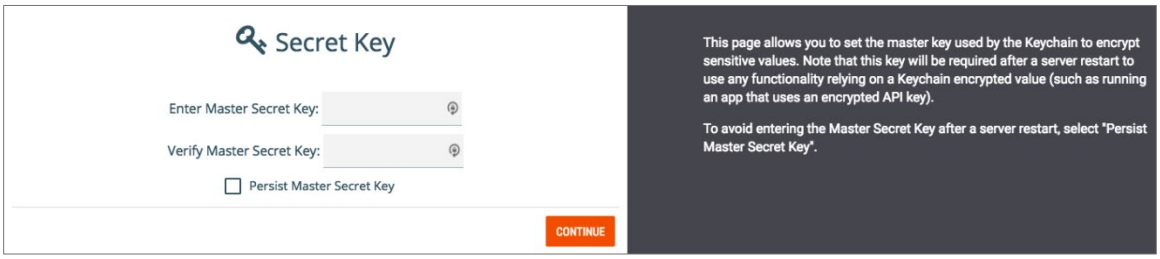

Figure 5

The Master Secret Key is used to encrypt sensitive values and is required on every server restart unless the **Persist Master Secret Key** box is checked. If the Master Secret Key is not ready to be set at this time, the **Proceed to Dashboard** button will bypass this screen until the next System Administrator login; however, all functionality relying on the Master Secret Key will fail.

Installing the License

NOTE: This section is only applicable if a license was not correctly inserted into ThreatConnect during the installer's configuration.

1. Log in to ThreatConnect with a System Administrator account.

2. On the top navigation bar (Figure 6), place the cursor on the **Settings**  icon and the **Settings** menu will appear (Figure 7). Select **System Settings** and the **System Settings** screen will appear (Figure 8).

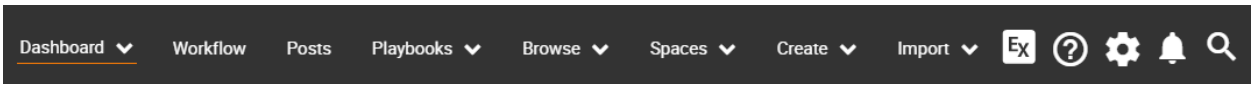

Figure 6

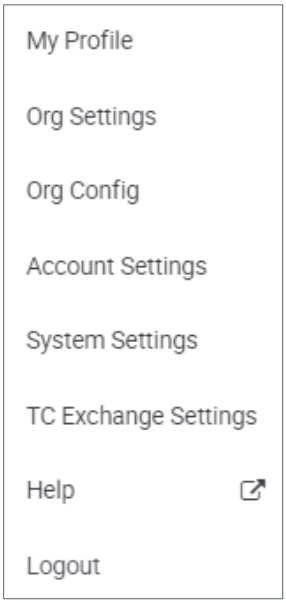

Figure 7

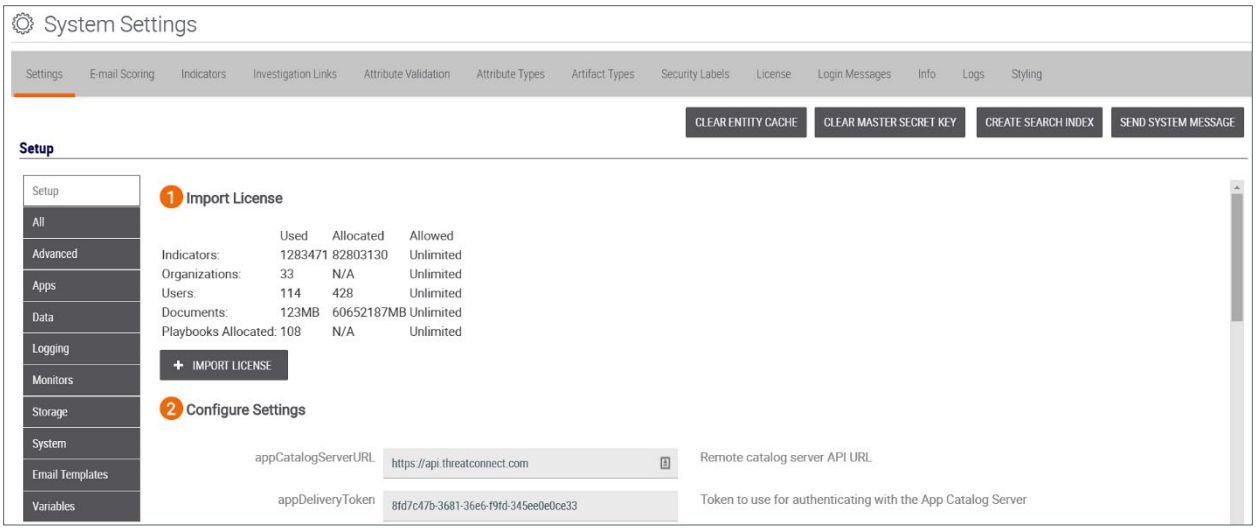

Figure 8

3. Click the **License tab**, and the **License** screen will appear with the **License Config** subtab highlighted (Figure 9), displaying the current allocations of Indicators, Organizations, Users, and Documents. From this screen, the user can also import a license.

Figure 9

NOTE: Certain users may be required to re-accept the terms of the License Agreement upon next login.

4. Click the **+ IMPORT LICENSE** button, and choose the License file from the directory.
5. Specific users may then be prompted to accept the license terms.
6. Restart ThreatConnect.

NOTE: Without a valid license, ThreatConnect will not function properly.

Dedicated Cloud and On-Premise System-Settings Checklist

Users should review the settings of their instance to ensure that it is configured according to their needs. Table 12 is a checklist showing the core properties and their values, assuming a standard ThreatConnect setup.

Table 12

Property	Common Value	Required For
appsApiUrl	https://threatconnect.customer.com/api	API URL for App Jobs
appsJavaHome	/usr/java/latest	TC Exchange Apps
appsPythonHome	/usr/bin	TC Exchange Apps
batchApiEnabled	true	Batch API
bulkIndicatorEnabled	true	Bulk Indicator Export
CALEnabled (four different CAL settings)	true	Collective Analytics Layer
dnsEnabled	true	DNS Data Services
documentStorageType	LOCAL or AWS	Document Storage
elasticSearchEnabled	true (Refer to *Configuring Elasticsearch on ThreatConnect* section.)	Elasticsearch
emailEnabled	true	Email Notifications and Alerts
ipGeoEnabled	true	IP GEO
keychainEnabled	true (Refer to *Setting Master Key for Keychain* section.)	App Credential Encryption
logToFile	true	App Logging
mailInboundEnabled	true	Feed and Phishing Mailboxes
reverseWhoisEnabled	true	Reverse Whois Data Services
secureSystemUrl	https://hostname	Images and Links in Emails and Posts

sourceFeedMonitorEnabled	true	HTTP Source Feeds
systemDisplayName	your_hostname	Email Recognition
systemEmailAddressAccount	your_sysadmins_email	Account Questions
systemEmailAddressNotification	your_sysadmins_email	Notification Questions
systemSubjectName	your_instance_name	Instance Recognition
systemUrl	http://hostname	Images and Links in Emails and Posts
taskEmailMonitorEnabled	true	Workflow Task Emails
taxiiExchangeMonitorEnabled	true	STIX/TAXII Feeds
threatAssessMonitorEnabled	true	ThreatAssess
threatDeprecationMonitor	true	Indicator Deprecation
whoisEnabled	true	Whois Data Services

Configuring Elasticsearch on ThreatConnect

1. Log in to ThreatConnect with a System Administrator account.
2. On the top navigation bar (Figure 6), place the cursor on the Settings  icon and the Settings menu will appear (Figure 7). Select System Settings and the System Settings screen will appear (Figure 8).
3. Select **ALL** in the vertical column on the left. Configure the settings **in this exact order:**
   a. **documentStorageType**: Value = AWS or Local. (Review information on Document Storage in the Threatconnect Account Administration Guide and the ThreatConnect System Administration Guide.)
   b. **elasticSearchUrl**: Value = URL
   c. Scroll down and click **Save Settings.**
   d. **logToElasticSearch**: Value = true (box is checked)

e. Scroll down and click **Save Settings.**

f. **elasticSearchEnabled**: Value = true (box is checked)

g. Scroll down and click **Save Settings.**

4. Click the **CREATE SEARCH INDEX** button, and the **Search Index Configuration** pop-up screen will appear (Figure 10) with the **Setup** tab highlighted. The top checkbox will index items currently existing in the ThreatConnect database. The bottom checkbox will index objects currently existing in document storage.

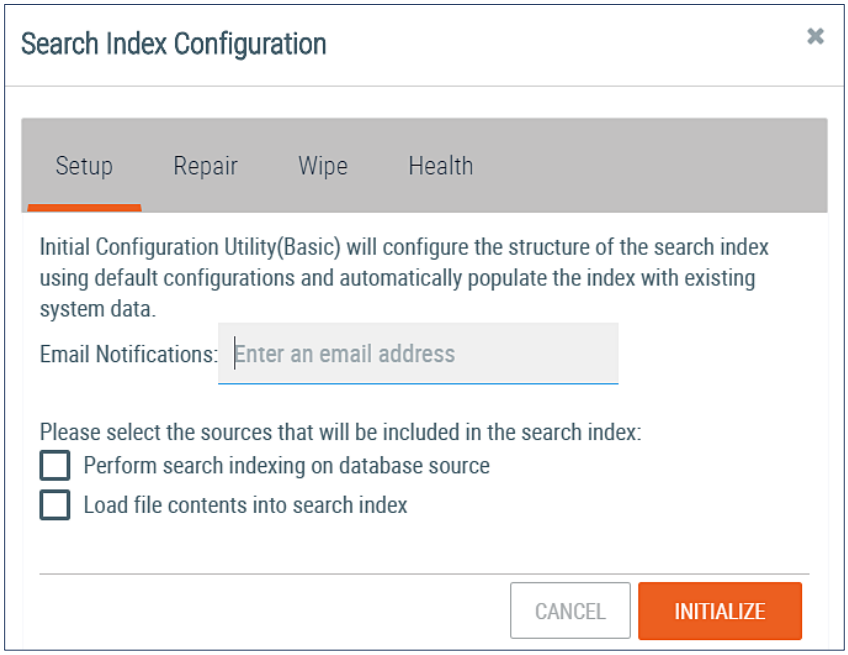

Figure 10

NOTE: The high availability of the index is not critical due to the amount of data. The index can be recreated at a rate of ~300,000 records/10min.

5. The **Repair** and **Wipe** tabs should **only** be accessed by **advanced** users, as these utilities will delete existing data.
6. Click the **Health** tab to view the health and status of the Elasticsearch nodes, index, and cluster.
7. Return to the **Setup** screen, check both boxes and click the **INITIALIZE** button.
8. The indexing status will be logged in the log file specified for ThreatConnect. Review the log by accessing the **Logs** tab of the **System Settings** screen.

Creating an Organization

The System Organization account cannot be used for creating Indicators or performing analyses—its sole job is system configuration and user administration. To generate additional users, add Indicators, etc., an Org (Organization) must be created. Please see the *ThreatConnect Organization Administration Guide* for information on creating and customizing Orgs.

APPENDIX A: POSTGRESQL INSTALLATION AND CONFIGURATION

This section will outline how to configure the PostgreSQL server and database to prepare it for usage with ThreatConnect, instead of using the MySQL database.

PostgreSQL Download and Installation

To acquire the version needed, download and install a file that has the distribution repository configured.

```
RedHat 7: yum install -y
https://download.postgresql.org/pub/repos/yum/11/redhat/rhel-7-ppc64le/pgdg-redhat11-11-2.noarch.rpm

CentOS 7: yum install -y
https://download.postgresql.org/pub/repos/yum/11/redhat/rhel-7-ppc64le/pgdg-centos11-11-2.noarch.rpm
```

NOTE: The installation file and its location may change, based on the application provider. Please consult Customer Service if the link does not function.

With the repository now installed and active, utilize **yum** to install the PostgreSQL server:

```
# yum install postgresql11 postgresql11-server -y
```

Once the installation is complete, initialize the database setup:

```
/usr/pgsql-11/bin/postgresql-11-setup initdb
```

Once the database setup is complete, enable on startup and start the PostgreSQL service:

```
# systemctl enable postgresql-11
# systemctl start postgresql-11
```

Port Configuration

To allow access to the PostgreSQL database from other servers, the firewall must be set up to allow incoming and outgoing connections:

```
firewall-cmd --permanent --zone=public --add-service=postgresql
firewall-cmd --reload
```

Alternative methods may be used for the port configuration, but port 5432 needs to be open to allow servers to connect and relay data to the PostgreSQL database.

PostgreSQL Access Configuration

By default, PostgreSQL is not set up with a password. A password will be needed in order to change the authentication setting to force a password prompt for any access to the PostgreSQL console or run scripts. To add a password to the postgres account:

```
su - postgres
psql -c “ALTER USER postgres WITH PASSWORD ‘<new-password>’”;
```

This will change the password for the **postgres** account in the PostgreSQL application to what was inserted in between the single quotes.

To allow access to the PostgreSQL server and databases, incoming connections need to be defined. The settings suggested below will force all local connections to the PostgreSQL use MD5 authentication (i.e. passwords), as well as define the same settings for the IP address(es) that will connect to the PostgreSQL server. The default location of this file is: **/var/lib/pgsql/11/data/pg_hba.conf:**

```
#Line add to file:
host    all         <IP-address-or-CIDR-block> md5

#Line change in file:
local   all     all                         peer

to:
local   all     all                         md5

#Line change in file:
host    all     all             127.0.0.1/32            ident

to:

host    all     all             127.0.0.1/32            md5
```

PostgreSQL Configuration Options

ThreatConnect requires the following options to be configured in the PostgreSQL configuration file that is default located **/var/lib/pgsql/11/data/postgresql.conf**:

```
listen_addresses = '<ip-of-PostgreSQL-server>'
synchronous_commit = off
effective_cache_size = 75% total memory available on server
shared_buffers = 25% of the effective_cache_size variable
work_mem = 32MB
max_parallel_workers = total CPU cores installed on the server -1
max_parallel_workers_per_gather = 3
max_worker_processes = total CPU cores installed on the server -1
```

Save the file and restart the postgresql-11 service to commit the changes made in both configuration files:

```
systemctl restart postgresql-11
```

Creating the Users and the Database

Once the PostgreSQL database service is installed and running, log in to create a database and user for ThreatConnect. From the command line on the PostgreSQL database server:

```
su - postgres
psql -U postgres -W
```

When prompted, enter the postgres password chosen upon creation.

A separate non-root user should be created for security reasons. In this example, the username **tcuser** and password **Password1!** were chosen. Enter the command below to create a new user:

```
CREATE ROLE tcuser WITH LOGIN PASSWORD 'Password1!';
```

Enter the command below to create a new database. In this example, the database name **"threatconnect"** and the user account that ThreatConnect application will use **"tcuser"** was chosen.

```
CREATE DATABASE "threatconnect" WITH OWNER "tcuser" ENCODING 'UTF8' LC_COLLATE =
'en_US.UTF-8' LC_CTYPE = 'en_US.UTF-8';
```

NOTE: Any database name and user account may be used, but it will be referenced later in the installation and configuration process.

Grant permissions on the newly created user, allowing them to access the tables within this database. Enter the following commands to log in to PostgreSQL and add privileges to the new user:

```
GRANT ALL PRIVILEGES ON DATABASE threatconnect TO tcuser;
```

NOTE: The above example assumes that the database was named threatconnect ***and the user*** tcuser**.**

Creating ThreatConnect Tables

Before starting on this step, download the ThreatConnect binary zip file and have it extracted. The **threatconnect/app/scripts/postgres** directory contains a **.sql** script to create the initial tables for ThreatConnect within the database just defined. Browse to the folder containing the threatconnect-**<version>.sql** file, run the command below and enter the root password again when prompted.

NOTE: The install sql script will use the default "public" schema under the database.

```
su - postgres
psql -U tcuser -d threatconnect
</opt/threatconnect/app/scripts/postgres/ThreatConnect-<version>sq
```

NOTE: This command specifies "threatconnect" ***as the name of the database. If a different name for the above database was chosen, use that instead.***

APPENDIX B: RE-AUTHENTICATING THE KEYSTORE AND SSL CERTIFICATE CONFIGURATION

Re-Authenticating Keystore

ThreatConnect requires the use of HTTPS and uses Secure Sockets Layer (SSL) certificates to guarantee security to users. By default, ThreatConnect is pre-configured with a keystore containing a self-signed SSL certificate. For the sake of authenticity, users may wish to replace this certificate with their SSL certificate. This option in the installer, then, allows a user to re-authenticate if the keystore is changed. Please note that ThreatConnect requires that the following assumptions are met:

- There is a Java keystore **.jks** file located at **config/keystore.jks**.
- This keystore contains a key pair with alias labeled "tc," without quotes.
- The password for this keystore has been entered via the **Re-authenticate SSL keystore password** option in the setup script.
- The keystore password AND the key password need to be the same.

Directions for configuring an additional keystore are below. Note that there are many ways to generate a keystore; the examples in the next sections use keytool as it is included with the JDK software listed in the *Software* section.

SSL Certificates

Generating a Self-Signed SSL Certificate

To use a new keystore with a self-signed certificate, refer to the keytool commands in this section. First, rename the existing **keystore.jks** in case it is needed later. Then run the following command in the **config** directory:

```
# keytool -genkeypair -alias tc -keyalg RSA -keystore keystore.jks -storepass
yourPassword --dname "CN=threatconnect"
```

Do not enter a key password for the **<tc>** alias when prompted. **Leave this field blank,** and press **enter** to confirm the use of the keystore password. (ThreatConnect and its dependent packages rely upon this behavior in the keystore.) This will generate a keystore named **keystore.jks**, with an alias of "tc" and a password of "yourPassword." This password will need to be entered into the setup script as detailed above to meet the three invariants specified in this section.

Generating a CA-Signed SSL Certificate

If a certification authority (CA) certificate will be installed on the ThreatConnect server, a new keystore must be created. To create a new keystore, log in to the ThreatConnect server, and access a directory where these files can be stored without impacting ThreatConnect (i.e.,/home or /opt). Run this command to create a new keystore:

```
# keytool -genkey -alias server -keyalg RSA -keysize 2048 -keystore
keystore.jks
```

A series of prompts will be given:

- Enter keystore password: <entered by the client>
- Re-enter new password: <entered by the client>
- What is your first and last name? <This needs to be the FQDN or DNS name that will be used to access the server.>
- What is the name of your organizational unit? <entered by the client>
- What is the name of your organization? <entered by the client>
- What is the name of your city or locality? <entered by the client>
- What is the name of your state or province? <entered by the client>
- What is the two-letter country code for this unit? <entered by the client>

A prompt will appear to ensure that all data entered is correct. Type **yes** if no changes need to be made.

- Enter key password for <server>: <Press **return** if same as keystore password.>
- <Press **enter**.>

Once the keystore is made, run this command to create the alias needed for ThreatConnect:

```
# keytool -changealias -keystore keystore.jks -alias server -destalias tc
```

To obtain a keystore that contains a certificate signed by a CA, generate a certificate-signing request:

```
# keytool -certreq -keyalg RSA -alias tc –keystore keystore.jks -file
certreq.csr
```

This will generate a certificate-signing request file, **certreq.csr**. Present this to the CA, who will return a signed file.

Option 1: CA returns with .crt file

The user may be required to import the root CA certificate as well, to maintain the trust. To import root certificates, request them from a CA, and import them using the command below, where **Root.crt** is the name of the certificate file provided by the CA.

```
# keytool -import -keystore keystore.jks -alias RootCA -file Root.crt
```

The user may be required to import the intermediate CA certificate as well, to maintain the trust. To import intermediate certificates, request them from a CA, and import them using the command below, where **intermediate.crt** is the name of the certificate file provided by the CA.

```
# keytool -import -keystore keystore.jks -alias intermediateCA -file
intermediate.crt
```

Once the other certificates are imported, import the file below back into the keystore, where **server.crt** is the name of the file returned by the CA.

```
# keytool -import -alias tc -keystore keystore.jks -file server.crt
```

Option 2: CA returns with .pem file

If the CA returns with **.pem** files, convert them for use in the keystore. Upload the certificates to the ThreatConnect server, and import them using the command below, where **intermediate_cert.pem** is the name of the certificate file provided by the CA, and **signed_cert.pem** is the name of the server certificate.

```
# openssl crl2pkcs7 -nocrl -certfile signed_cert.pem -out cert.p7b -certfile
intermediate_cert.pem
```

This action will create a new file named **cert.p7b**. This file will be used to import into the keystore with this command:

```
# keytool -import -alias tc -keystore keystore.jks -trustcacerts -file
cert.p7b
```

Option 3:– CA returns with .cer file

If the CA returns with **.cer** files, ensure that the root CA and intermediate CA files have been returned for import. If the root and intermediate certificates are not available, the keytool cannot maintain the trust on the signed certificate. Start by copying all certificate files to the ThreatConnect server and importing the root CA certificate, where **rootca.cer** is the name of the root certificate file:

```
# keytool -import -alias RootCA -keystore keystore.jks -trustcacerts -file
rootca.cer
```

Continue with the intermediate CA certificate, where **intermediateca.cer** is the name of the intermediate CA certificate file:

```
# keytool -import -alias IntermediaCA -keystore keystore.jks -trustcacerts -
file intermediateca.cer
```

Finally, add the certificate for the ThreatConnect server, where **tc_cert.cer** is the name of the server certificate:

```
# keytool -import -alias tc -keystore /path/to/keystore.jks -trustcacerts -
file tc_cert.cer
```

After one of the options is completed, a new keystore will be ready to be used by ThreatConnect. To use the new keystore, first, stop the ThreatConnect service. Rename the **keystore.jks** file in /opt/threatconnect/config to **keystore.jks.original**. Copy the newly created **keystore.jks** file to **/opt/threatconnect/config**/. Be sure to change ownership on the new **keystore.jks** file by running this command:

```
# chown threatconnect:threatconnect /opt/threatconnect/config/keystore.jks
```

Next, the certificates need to be added to the JRE keystore located at **/usr/java/jdk<version>/jre/lib/security/**:

```
# cd /usr/java/jdk<version>/jre/lib/security/
```

Root CA:

```
# keytool -import -alias RootCA -keystore cacerts -trustcacerts -file
<rootCA_filepathandname>
```

Intermediate CA:

```
# keytool -import -alias IntermediateCA -keystore cacerts -trustcacerts -file
<intermediateCA_filepathandname>
```

Server CA:

```
# keytool -import -alias tc -keystore cacerts -trustcacerts -file
<serverCA_filepathandname>
```

Next, log in as the threatconnect user:

```
# su – threatconnect
```

Change the current location to /opt/threatconnect/app/:

```
# cd /opt/threatconnect/app/
```

Run **setup.sh**:

```
# ./setup.sh
```

Select the option to change the SSL password. Enter the password twice that was created for the new keystore. Log out as the **threatconnect** user and start the **threatconnect** service. Once the **threatconnect** service is fully started, log in to the Web interface of ThreatConnect. Verify that the certificate is found by the browser and that SSL is working.

APPENDIX C: CONFIGURING SSL TRANSMISSION WITH MYSQL AND THREATCONNECT

This section will guide the user through enabling SSL transmission between the MySQL server and the ThreatConnect application. This configuration is an option setting and not a requirement for ThreatConnect to function.

First, the SSL certificates for the MySQL server will need to be located and utilized. Based on the default installation of MySQL defined in this document, the installation process will have autogenerated the SSL certificates for MySQL to use. The certificates must be saved as a **.pem** format for MySQL.

Update MySQL server configuration (**/etc/my.cnf**) with the following lines.

NOTE: These settings need to be placed at the end of the** [mysql] **section.

```
#SSL Settings
ssl-ca=/var/lib/mysql/ca.pem
ssl-cert=/var/lib/mysql/server-cert.pem
ssl-key=/var/lib/mysql/server-key.pem

[client]
ssl-ca=/var/lib/mysql/ca.pem
ssl-cert=/var/lib/mysql/server-cert.pem
ssl-key=/var/lib/mysql/server-key.pem
```

The above paths and keys are the autogenerated ones from the installation of MySQL. If these are not to be used, simply substitute the path and filenames.

Once the configuration is updated, save and close the file. The service will need to be restarted to apply the settings:

```
systemctl restart mysqld
```

Login to the MySQL console and run the following command:

```
status;
```

The status output will show the settings used in the connection. SSL will be defined and the SSL settings that are default set.

The file defined as **ssl-cert** in the **[client]** section of the SSL Settings will need to be copied to the ThreatConnect application server. The certificate will need to be installed in the ThreatConnect keystore. The command to import the file is:

```
keytool -import -alias <IP-or-FQDN-of-MYSQL-server> -keystore
/opt/threatconnect/config/keystore.jks -file <path-to-server-cert-file>
```

NOTE: The above command assumes the default installation of ThreatConnect in** /opt/threatconnect. **Adjust if the installation path is different.

There will be a prompt for the password to the keystore. Once the password is entered, type **yes** to save the certificate into the keystore.

The process will need to be duplicated into the Java keystore on the ThreatConnect application server. Run the following command:

```
keytool -import -alias <IP-or-FQDN-of-MYSQL-server> -keystore
/usr/java/latest/jre/lib/security/cacerts -file <path-to-server-cert-file>
```

The password for the keystore is **changeit** . Type **yes** to save the password into the keystore.

Once the certificate has been imported, the ThreatConnect configuration will need to be updated to use the SSL connection when communicating with MySQL. The default location for the file is located at: **/opt/threatconnect/config/threatconnect.xml.**

Change two lines in the config. They both look like this:

```
<connection-
url>jdbc:mysql://mysql.example.com:3306/threatconnect?autoReconnect=true&amp;
rewriteBatchedStatements=true</connection-url>
```

Update the lines to include the SSL setting. For example:

```
<connection-
url>jdbc:mysql://192.168.1.22:3306/threatconnect?autoReconnect=true&amp;rewri
teBatchedStatements=true&amp;useSSL=true</connection-url>
```

Save the configuration file.

Start the ThreatConnect service or continue the installation process defined in the ThreatConnect installation section.

Verify that the application is using SSL to connect by connecting to the MySQL server and running the following command:

```
tcpdump -l -i <name-of-nic> -w - src or dst port 3306 | strings
```

The output from the above command should just be garbled text. If the transmission between the servers is legible, run through Appendix C again to ensure no configurations or commands were missed.