Trapeze Platform Version 3

Deployment requirements

Trapeze 3 is deployed in a variety of configurations ranging from single server installation for non-critical use cases (including pilots) to distributed services using multiple servers to support high availability installations.

Many factors influence the server capacity requirements such as number of devices, event observation rate, nature of event processing, and integration touchpoints. The scenarios covered in this document are general scenarios, and they may differ for specific installations.

This document outlines the minimum requirements for individual server installations.

Server specifications

The values in the following table are recommended for physical or virtual servers. In the case of deployment on multiple servers, they should preferably be provisioned in separate availability zones.

Component	**Minimum Requirement**
Computing	Virtual or Physical Machine with Pentium Compatible CPU that is: ● 1.4 GHz or higher ● 4-8 cores Computers based on CPUs that are compatible with the AMD64 (x86- 64) and Extended Memory 64-bit Technology (EM64T) processor architecture are considered x64-based systems. Trapeze can utilize all the cores available in the system.
Memory	16-32 GB of RAM Depending on the workload (concurrent users and integrations) and nature of long-lived event processing queries, larger memory may be required.
Storage	100 GB (or more) of available disk space including the operating system, pre-requisites, log space and all software. Recommended secondary disks for ephemeral storage such as caching, and logs
Operating system	Ubuntu 22.04.3 LTS or higher LTS version An administrative account (sudo access with ssh) is needed for installation and maintenance. User should be able to access docker CLI and execute ansible tasks with *become flag* enabled.
Dependencies for deploy	Docker version 24.0.7 or higher Docker_packages: docker-ce, docker-ce-cli, containerd.io, docker-compose-plugin Ansible version: core 2.15.6 Ansible packages: ansible, sshpass, python3-apt, python3-pip, python3-docker Ansible galaxy community modules: community.general, community.docker, community.crypto

Deployment procedure uses Ansible for initial setup and maintenance. If the user with sudo privileges is not available, maintenance / deployment can still be performed if prerequisites listed above are installed and require a dedicated user for running setup and maintenance that is a member of docker group.

Network / Environment Specifications

All servers must be assigned fixed IP addresses.

Firewall

All servers should be deployed on a network behind a security group to allow for east – west traffic between the nodes. Specifically, the following ports should be open:

Docker Swarm and Machine Access Ports:

Port	Protocol	Description
2377	TCP	Cluster management communications (HA ONLY)
7946	TCP, UDP	Inter-node communications (HA ONLY)
4789	UDP	Overlay network traffic
5084	TCP	LLRP
1883	TCP	MQTT
8883	TCP	MQTT with TLS
80	TCP	Trapeze Web API & UI
443	TCP	SSL port for Trapeze Web API & UI
22	TCP	SSH port for operating system shell

Consul Service Access Ports (HA ONLY):

Port	Protocol	Description
8600	TCP, UDP	DNS
8500	TCP	HTTP
8503	TCP	GRCP
8300	TCP	Server RPC
8301	TCP, UDP	LAN serf
8302	TCP, UDP	WAN serf

Internet access

All servers need access to software and configuration repositories for installation, and the firewalls must be configured to allow access to the following.

InThing's repositories at:

- https://dev.azure.com
- inthing.azurecr.io

Ansible repositories at:

- ppa.launchpadcontent.net
- galaxy.ansible.com
- pypi.org

If a security policy around east-west traffic encryption is mandatory, it is possible to deploy the cluster on an encrypted IPsec overlay network (ESP). Encrypted communication does increase the load on the machines and might impact the event handling capacity of the cluster.

DNS – Multiple servers

A DNS record that resolves all of the node IP addresses needs to be created (round-robin DNS). This is simplest way to ensure ingress traffic to Trapeze UI and apps running in Trapeze remain uninterrupted if one of the nodes becomes unavailable.

TLS – Multiple servers

Ingress to Trapeze UI and other apps is possible via TLS encrypted connection. Once a round-robin DNS record is in place, a TLS certificate with FQDN (fully qualified domain name) containing that DNS name needs to be provided in OpenSSL format.

When Trapeze is deployed on-prem, it is not publicly available so generally trusted CAs (certificate authorities) cannot provision TLS certificates as they cannot perform DNS name validation. If a client uses their own CA, it should be used to sign TLS certificates provided to be used with Trapeze. Alternatively, a self-signed TLS certificate will be generated by InThing which will trigger a warning in the browser and/or other clients accessing Trapeze.

Deployment Checklist / Information sheet:

Below table should be populated with the information applicable to the network/application setup.

Environment		
VPN / direct access	*requirement*	Initial setup and maintenance
User permissions	*requirement*	**sudo** access is required
CPU 4-8 cores	*requirement*	
Memory 16-32 GB	*requirement*	
Storage 100 GB (or more)	*requirement*	
Server		
FQDN (URL)	*host.example.domain*	used for TLS certificates
node 1 IP address	*172.129.1.10*	
node 2 IP address	*172.129.1.11*	
node 3 IP address	*172.129.1.12*	
network	*172.129.1.0/24*	
gateway	*172.129.1.1*	
Trapeze		
Providers	*manufacturer / model*	reader device specific

Streaming Apps	*Use case / business logic / custom*	use case specific
Integration connectors	*integration to Visium or other enterprise applications*	use case specific
Display apps	*visualization tool*	use case specific