What is the Docker Store?

The Store is the best way for you to distribute and sell your Dockerized content. Publish your software through the Docker Store to experience the benefits below:

1. Access to Docker's large and growing customer-base. Docker has experienced rapid adoption, and is wildly popular in dev-ops environments. Docker users have pulled images over four billion times, and they are increasingly turning to the Docker Store as the canonical source for high-quality, curated content.

2. Customers can try or buy your software, right from your product listing. Your content is accessible for installation, trial, and purchase from the Docker Store and the Docker CLI.

3. Use our licensing support. We can limit access to your software to a) logged-in users, b) users who have purchased a license, or c) all Docker users. We'll help you manage and control your distribution.

4. We'll handle checkout. You don't have to set up your own digital e-commerce site when you sell your content through the Docker Store. We'll even help you set pricing—and you can forget about the rest.

5. Seamless updates and upgrades. We tell customers when your content has upgrades or updates available, right inside their Docker host product.

6. It's a win-win for our platform and publishers: great content improves our ecosystem, and our flexible platform helps you bring your content to market.

7. Achieve the Docker Certified quality mark. Publisher container images and plugins that meet the quality, security, and support criteria of the program will display a "Docker Certified" badge within the Docker Store and external marketing.

Distribution Models

The Docker Store welcomes free and open-source content, as well as software sold directly by publishers. We support the following commercial models:

Paid-via-Docker Content

This is content for which customers transact via Docker, as described in the publisher agreement. Paid-via-Docker content includes both software than can be deployed on a host, as well as software that runs in the cloud and can be accessed by the customer via an ambassador container (containerized cloud services, for example).

Free Content

Free content is provided free-of-charge, and customers may pull it from the Docker Hub either at their discretion or upon license acceptance, at the publisher's discretion. You agree that you will not charge customers for any Free Content by making it available for purchase outside of the Docker Store.

Publishing Content on the Docker Store

Permitted Content and Support Options

- Content that runs on a Docker Enterprise Edition (i.e. Docker Certified Infrastructure) may be published in the Store. This content may also qualify to become a Docker Certified Container or Plugin image and be backed by collaborative Docker/Publisher support
- Content that runs on the Docker Community Edition may be published in the Store, but will not be

supported by Docker nor is it eligible for certification.

- Content that requires a non Certified Infrastructure environment may not be published in the Store.

If your Content:	Can Publish on Store	Can be Certified and Supported by Docker	Supported by Publisher
Works on Docker Enterprise Edition	YES	YES	Required
Works on Docker Community Edition	YES	NO	Optional
Does not work on Docker Certified Infrastructure	NO	NA	NA

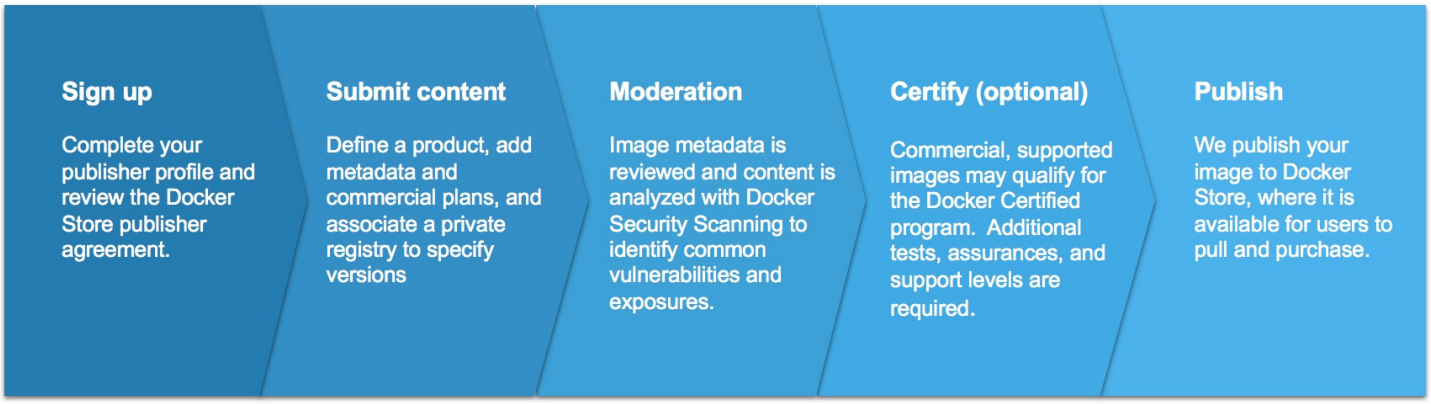
Onboarding

The publishing process for the Docker Store is straightforward, and can be initiated from the landing page. You can sign in with your Docker ID, and specify a product name and image source from a private repository. We require that your product images are stored in private repositories via Docker Cloud and/or Hub, as they serve as an internal staging area from which you can revise and submit content for review.

Once you specify a private-repository source for your product, you can provide the content-manifest items to populate your product's details page. These items include logos, descriptions, and licensing and support links so that customers can make informed decisions about your image. These items are submitted alongside the image itself for moderation.

The Docker Store team then conducts a comprehensive review of your image and metadata. We use Docker Security Scanning to evaluate your product images' security, and share results with you as the publisher. During the image-moderation phase, we iterate back and forth with publishers to address outstanding vulnerabilities and content-manifest issues until the image is ready for publication.

Commercial content and other supported images may qualify for the Docker Certified Container or Plugins quality mark. The testing for this program goes beyond the vulnerability scan and also evaluates container images for Docker best practices developed over years of experience. Collaborative support capability between Docker and the publisher is also established. Please refer to the diagram below for a high-level summary:



Create Great Content

Create your content, and follow our best practices to Dockerize it. Keep your images small, your layers few, and your components secure. Please refer to the links and guidelines listed below to build and deliver great content:

- https://docs.docker.com/engine/userg...est-practices/ (https://docs.docker.com/engine/userguide/eng-image/dockerfile_best-practices/)
- https://docs.docker.com/docker-hub/official_repos/ (https://docs.docker.com/docker-hub/official_repos/)
- https://github.com/docker/docker-bench-security (https://github.com/docker/docker-bench-security)

Here are some best practices when it comes to building vulnerability-free Docker images:

Choose a Secure Base Image (Dockerfile's FROM: directive)

Many base images have a strong record of being secure, including:

- Debian (https://hub.docker.com/r/library/debian/tags/jessie/) Linux: both small and tightly-controlled, Debian-linux is a good alternative if you're currently using Ubuntu.
- Alpine (https://hub.docker.com/_/alpine/) Linux: Alpine is a minimal linux distribution with an excellent security record.
- Alpine-based application images: these include python:alpine, ruby:alpine, and golang:alpine. They are secure and minimal, while providing the convenience of their non-Alpine alternatives.

Docker strongly recommends Alpine Linux. The founder of this Linux distribution is leading an initiative at Docker to provide safe, compact base images for all container applications.

Remove Unused Components

Often, vulnerabilities exist in components that aren't actually used in the containerized application. To avoid this, you can:

- Follow best practices when using the apt-get command.
- Make sure to run apt-get-remove to destroy any components required to build but not actually run your application. Usually, this involves creating multi-line Dockerfile directives, as seen below. The following example shows how to remove curl and python-pip after they are used to install the python requests package, all in a single Dockerfile directive:

```
RUN apt-get update && \
         apt-get install -y --no-install-recommends curl python-pip && \
         pip install requests && \
         apt-get remove -y python-pip curl && \
         rm -rf /var/lib/apt/lists/
```

Keep in mind: any file introduced in one directive of your Dockerfile can only be removed in the same directive (and not in subsequent directives in your Dockerfile).

Keep Required Components up-to-date

Your images are comprised of open-source libraries and packages that amass vulnerabilities over time and are consequently patched. To optimize your product’s integrity, you must keep your images up-to-date:

- Periodically update your base image's version, especially if you’re using a version deemed to be vulnerable.
- Re-build your image periodically. Directives including commands such as apt-get install ... pull the latest versions of dependencies, which may include security fixes.

Scan Your Own Private Repositories

Eliminating vulnerabilities is a trial-and-error process. To speed it up, consider using Docker Security Scanning on your own private Docker repositories in Docker Cloud and Docker Hub. This feature allows you to scan images you create on-demand, without relying on the scans provided by the Docker Publisher Program.

Create and Maintain Your Publisher Profile in the Store

Let the Docker community know who you are. Add your details, your company story, and what you do. At the very minimum, we require:

- Legal entity name
- Company website
- Phone number
- Valid company email
- Company icon/logo (square; at least 512x512px)

Prepare Your Image-manifest Materials

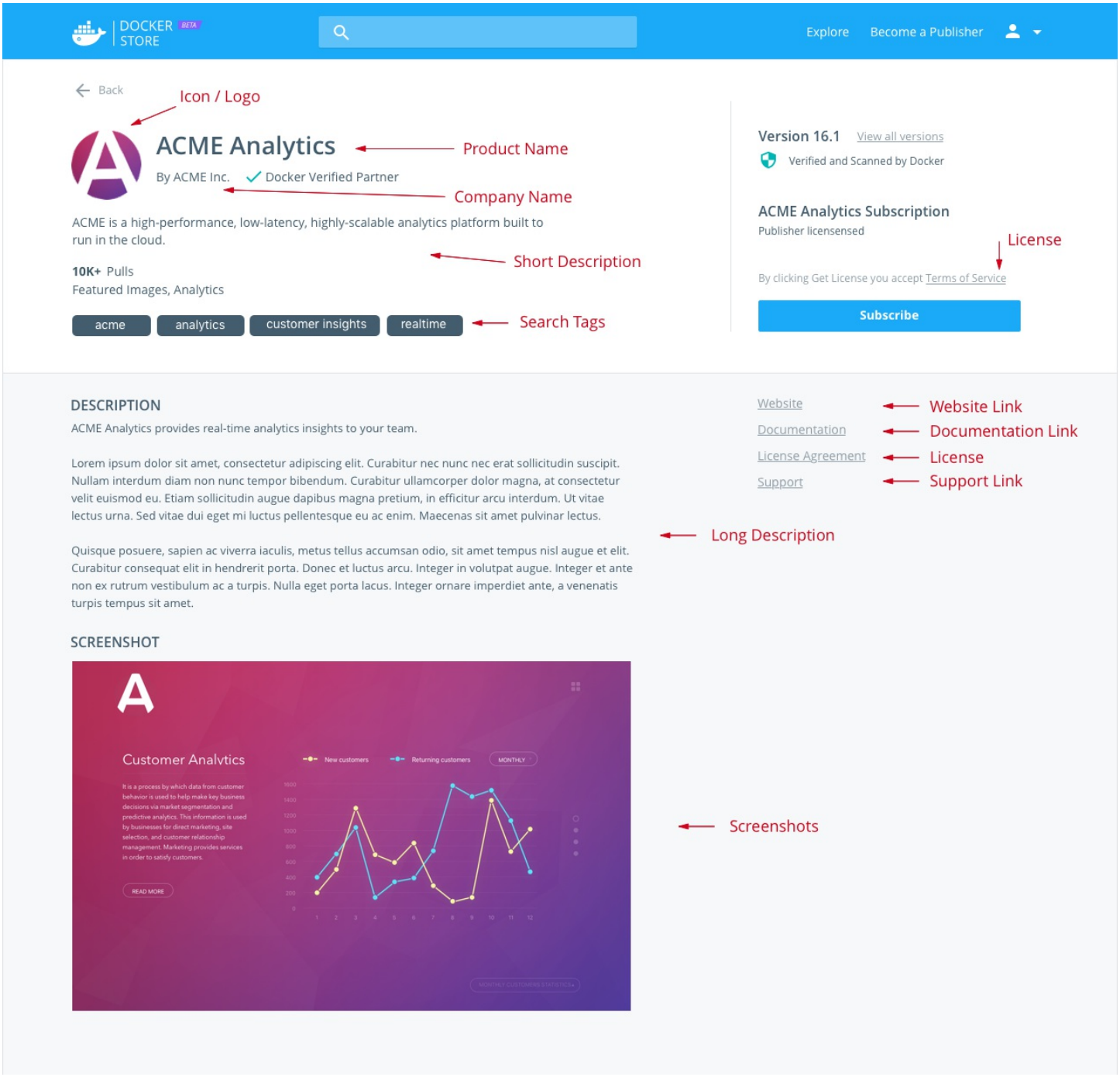
You must provide the namespace (including repository and tags) of a private repository on Docker Cloud or Hub that contains the source for your product. This repository path will not be shown to users, but the repositories you choose determine the Product Tiers available for customers to download.

The following content information helps us make your product look great and discoverable:

1. Product Name
2. Product icon/logo
3. Short description: a one-to-two-sentence summary; up to 140 characters
4. Category: Database, Networking, Business Software, etc. and any search tags
5. Long description: includes product details/pitch
6. Screenshot(s)
7. Support link
8. Product tier name
9. Product tier description
10. Product tier price
11. Installation instructions
12. Link to license agreements

How the Manifest Information is Displayed in the UI

(Please note that this is an approximate representation. We frequently make enhancements to the look and some elements might shift around.)



Support Your Users

Docker users who download your content from the Store might need your help later, so be prepared for questions! The information you provide with your submission will save support time in the future.

Support Information

If you provide support along with your content, include that information. Is there a support website? What email address can users contact for help? Are there self-help or troubleshooting resources available?

Support SLA

Include a Service Level Agreement (SLA) for each image you're offering for the Store. An SLA is your commitment to your users about the nature and level of support you provide to them. Make sure your SLA includes support hours and response-time expectations, where applicable.

Security and Audit Policies

Docker Security Scanning

We use Docker Security Scanning to automatically and continuously assess your products' integrity. The tool deconstructs images, conducts a binary scan of the bits to identify the open-source components present in each image layer, and associates those components with known vulnerabilities and exposures. We then share the scan results with you as the publisher, so that you can modify your images' content accordingly. Your scan results are private, and are never shared with end customers or other publishers.

To interpret the results, refer to the documentation (https://docs.docker.com/docker-cloud/builds/image-scan/#/view-docker-security-scanning-results).

Classification of Issues

- All Scan results will include the CVE numbers and a CVSS (Common Vulnerability Scoring System) Score.
  - CVE Identifiers (also referred to by the community as "CVE names," "CVE numbers," "CVE entries," "CVE-IDs," and "CVEs") are unique identifiers for publicly-known, cyber-security vulnerabilities.
  - The Common Vulnerability Scoring System (CVSS) provides an open framework for communicating the characteristics and impacts of IT vulnerabilities. Its quantitative model ensures repeatable, accurate measurement while enabling users to see the underlying vulnerability characteristics that were used to generate the scores. As a result, CVSS is well-suited as a standard measurement system for industries, organizations, and governments that need accurate and consistent vulnerability-impact scores. CVSS is commonly used to prioritize vulnerability-remediation activities, and calculate the severity of vulnerabilities discovered on systems. The National Vulnerability Database (NVD) provides CVSS scores for almost all known vulnerabilities.
- Docker classifies the severity of issues as:

CVSS Range	Docker Classification	SLA For Fixing the Issues
7.0 to 10.0	Critical	Within 72 hrs of notification
4.0 to 6.9	Major	Within 7 days of notification
0.1 to 3.9	Minor	No SLA. Best-effort to fix or address in documentation.

In addition to CVSS, the Docker Security team can identify or classify vulnerabilities that need to be fixed, and categorize them in the minor-to-critical range.

- The publisher is presented with initial scan results, including all components with their CVEs and their CVSS scores.
- If you use Docker's Scanning Service, you can subscribe to a notification service for new vulnerabilities.
- Failure to meet above SLAs may cause the listing is put on "hold".
- A warning label shows up on the marketplace listing. An email is sent to the users who have downloaded and subscribed for notifications.
- A Repo's listing can stay in the Hold state for a maximum of 1 month, after which the listing will be

revoked.

Usage Audit and Reporting

Unless otherwise negotiated, an audit of activity on publisher content will be retained for no less than 180 days.

A monthly report of said activity will be provided to the publisher with the following data: (1) report of content download by free and paid customers by date and time; (2) report of purchase, cancellations, refunds, tax payments, where applicable, and subscription length for paid customers of the content; and (3) the consolidated amount to be received by the publisher.

Certification

There are three types of certification that appear in Docker Store

Certifies that a container image on Docker Store has been tested; complies best practices guidelines; will run on a Docker Certified Infrastructure; has proven provenance; been scanned for vulnerabilities; and is supported by Docker and the content publisher.

This certification is designed for volume, network, and other plugins that access system level Docker APIs. Docker Certified Plugins provide the same level of assurance as a Docker Certified Container, but go further by having passed an additional suite of API compliance testing.

Indicates that the release of the Docker Edition and the underlying platform have been tested together and are supported in combination by both Docker and the partner.

Docker Certified Publisher FAQ

What is the Docker Certified program?

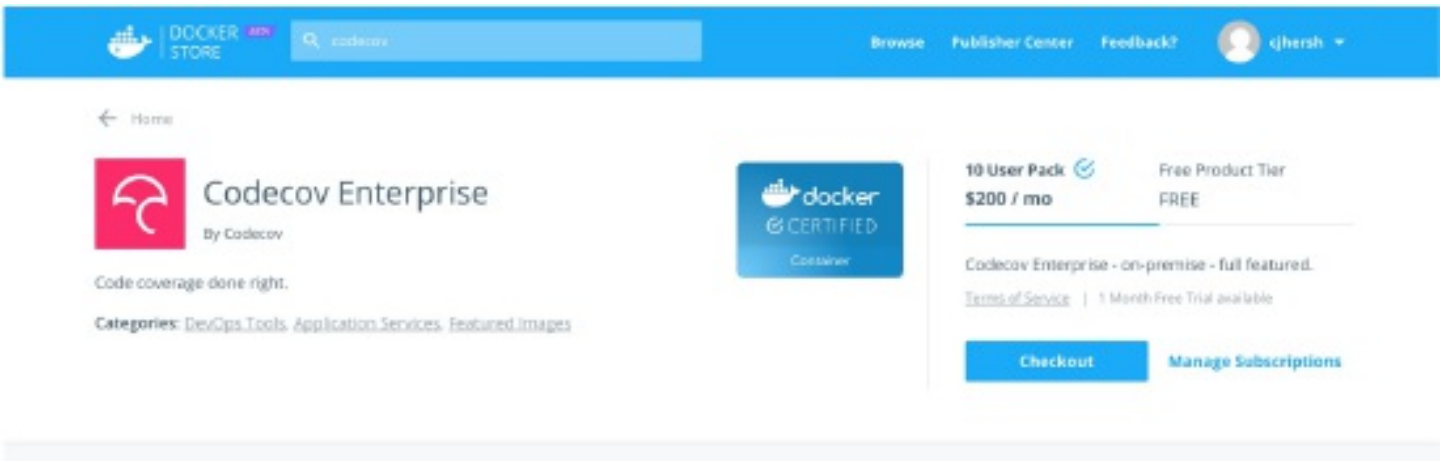
Docker Certified Container images and plugins are meant to differentiate high quality content on Docker Store. Customers can consume Certified Containers with confidence knowing that both Docker and the publisher will stand behind the solution. Further details can be found in the Docker Partner Program Guide (https://www.docker.com/partnerprogramguide).

What are the benefits of Docker Certified?

Docker Store will promote Docker Certified Containers and Plugins running on Docker Certified Infrastructure trusted and high quality content. With over 8B image pulls and access to Docker's large customer base, a publisher can differentiate their content by certifying their images and plugins. With a revenue share agreement, Docker can be a channel for your content. The Docker Certified badge can also be listed alongside external references to your product.

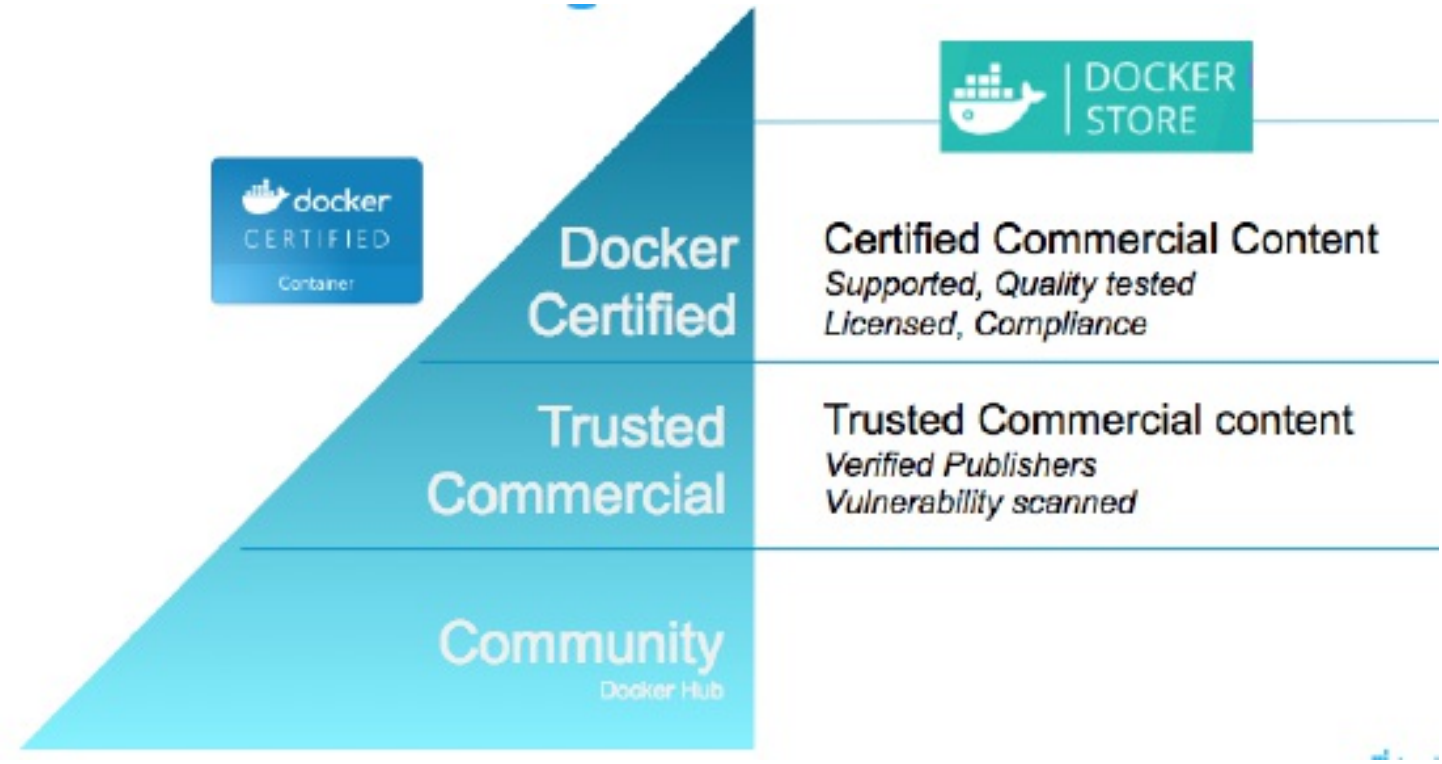
How will the Docker Certified Container image be listed on Docker Store?

These images are differentiated from other images on store through a certification badge. A user can search specifically for CI's by limiting their search parameters to show only certified content.



Is certification optional or required to be listed on Store?

Certification is recommended for most commercial and supported container images. Free, community, and other commercial (non-certified) content may also be listed on Docker Store.



How will support be handled?

All Docker Certified Container images and plugins running on Docker Certified Infrastructure come with SLA based support provided by the publisher and Docker. Normally, a customer contacts the publisher for container and application level issues. Likewise, a customer will contact Docker for Docker Edition support. In the case where a customer calls Docker (or vice versa) about an issue on the application, Docker will advise the customer about the publisher support process and will perform a handover directly to the publisher if required. TSAnet (https://www.tsanet.org/) is required for exchange of support tickets between the publisher and Docker.

How does a publisher apply to the Docker Certified program?

Start by applying to be a Docker Technology Partner (https://goto.docker.com/partners)

- Requires acceptance of partnership agreement for completion
- Identify commercial content that can be listed on Store and includes a support offering
- Test your image against the Docker CS Engine 1.12+ or on a Docker Certified Infrastructure version 17.03 and above (Plugins must run on 17.03 and above)
- Submit your image for Certification through the publisher portal. Docker will scan the image and work with you to address vulnerabilities. Docker will also conduct a best practices review of the image.
- Be a TSAnet (https://www.tsanet.org/) member or join the Docker Limited Group.
- Upon completion of Certification criteria, and acceptance by Docker, Publisher's product page will be updated to reflect Certified status.

Is there a fee to join the program?

In the future, Docker may charge a small annual listing fee. This is waived for the initial period.

What is the difference between Official Images and Docker Certified?

Many Official images will transition to the Docker Certified program and will be maintained and updated by the original owner of the software. Docker will continue to maintain of some base OS images and language frameworks.

How will certification of plugins be handled?

Docker Certification program recognizes the need to apply special scrutiny and testing to containers that access system level interfaces like storage volumes and networking. Docker identifies these special containers as “Plugins” which require additional testing by the publisher or Docker. These plugins employ the V2 Plugin Architecture that was first made available in 1.12 (experimental) and now available in Docker Enterprise Edition 17.03