National Institute of Standards and Technology
NIST

Information Technology Laboratory – Cybersecurity Program

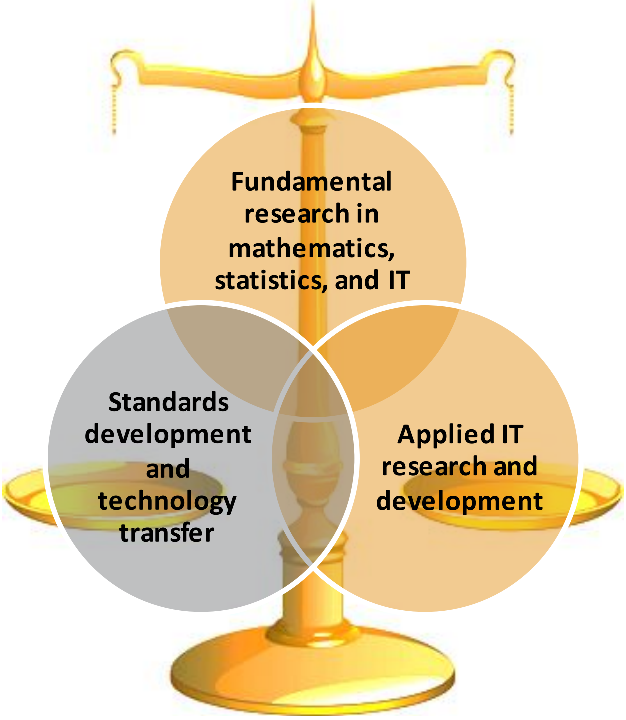
NIST Information Technology Laboratory – itl.nist.gov

Cultivating trust in IT and metrology through measurements, standards and tests

Cybersecurity Program

Standards and Guidelines Development – csrc.nist.gov

- Cryptographic Development – AES, SHA-3, PQC, etc.
- Cryptographic Validation – FIPS 140-2
- Risk Management Framework – Cybersecurity Framework, FISMA, SP 800-53, SP 800-171, etc.
- Technology Guidelines – Virtualization, Containers, Security Automation, etc.
- Privacy Framework
- Identity Management



Fundamental research in mathematics, statistics, and IT

Standards development and technology transfer

Applied IT research and development

National Cybersecurity Center of Excellence (NCCoE) – nccoe.nist.gov

Accelerate adoption of secure technologies: collaborate with innovators to provide real-world, standards-based cybersecurity capabilities that address business needs

DEFINE

ASSEMBLE

BUILD

ADVOCATE

Collaboration with Industry, Federal/State/Local Governments, and Academia

Application Container Security Guide

DevSecOps Use Case

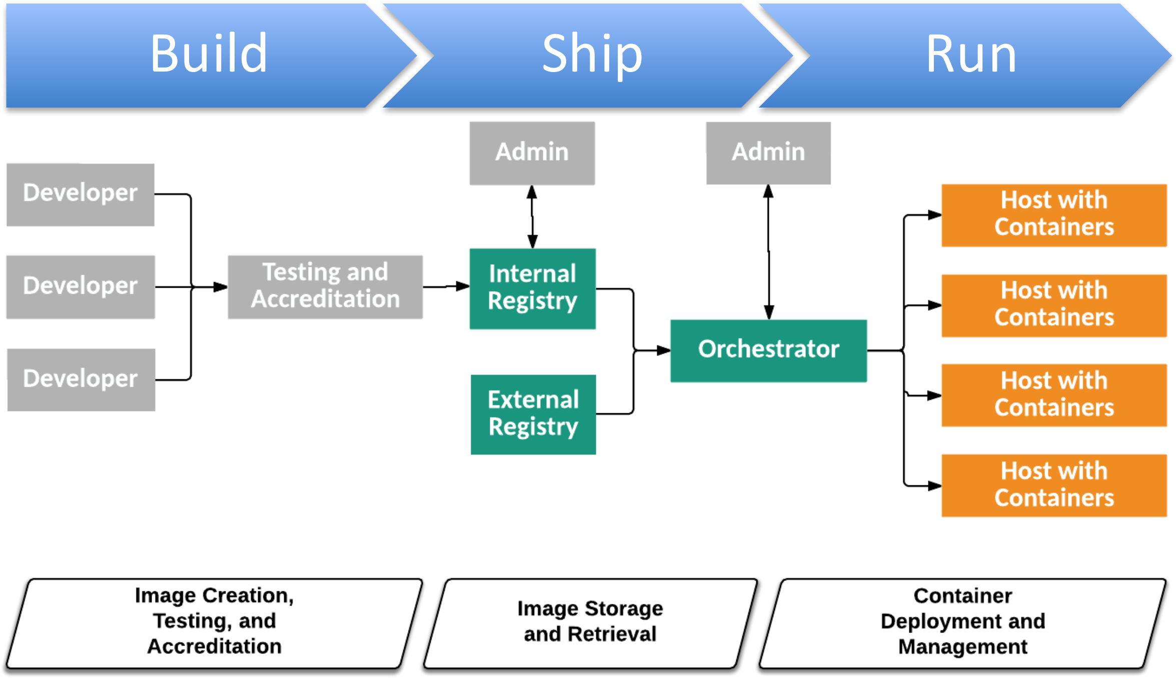
NIST Special Publication (SP) 800-190

https://csrc.nist.gov/publications/detail/sp/800-190/final

Abstract

Application container technologies, also known as containers, are a form of operating system virtualization combined with application software packaging. Containers provide a portable, reusable, and automatable way to package and run applications. This publication explains the potential security concerns associated with the use of containers and provides recommendations for addressing these concerns.

DevOps – Container Use Case

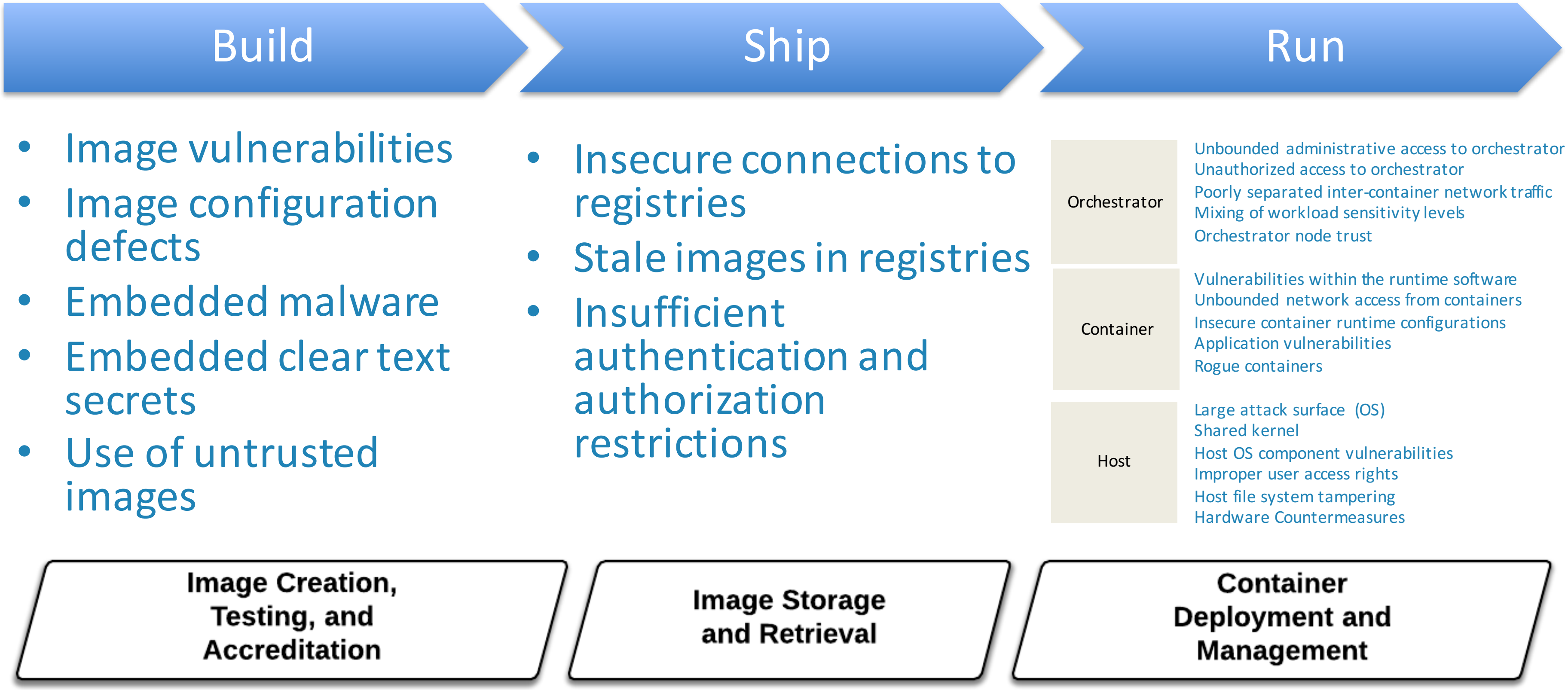


Modern container technologies have largely emerged along with the adoption of development and operations (DevOps) practices that seek to increase the integration between building and running apps, emphasizing close coordination between development and operational teams.

DevOps - Container Use Case

1. Image Creation, Testing, and Accreditation where you use some kind of an continuous integration tool to layer various binaries, libraries, and configuration files to build an image and test its functions and security posture.
2. Next we go into the Image Storage and Retrieval phase where images and associated meta-data are stored in central repositories called registries where developers can pull the images via APIs or the images are distributed to different hosts.
3. Finally, in the Container Deployment and Management phase, tools like orchestrators pull the images from the registries to deploy them into containers and manage the running containers.

DevSecOps – Container Use Case



Build

- Image vulnerabilities
- Image configuration defects
- Embedded malware
- Embedded clear text secrets
- Use of untrusted images

Image Creation, Testing, and Accreditation

Ship

- Insecure connections to registries
- Stale images in registries
- Insufficient authentication and authorization restrictions

Image Storage and Retrieval

Run

Orchestrator	Unbounded administrative access to orchestrator Unauthorized access to orchestrator Poorly separated inter-container network traffic Mixing of workload sensitivity levels Orchestrator node trust
Container	Vulnerabilities within the runtime software Unbounded network access from containers Insecure container runtime configurations Application vulnerabilities Rogue containers
Host	Large attack surface (OS) Shared kernel Host OS component vulnerabilities Improper user access rights Host file system tampering Hardware Countermeasures

Container Deployment and Management

Secure Software Development Framework (SSDF)

Mitigating the Risks of Software Vulnerabilities

NIST Cybersecurity White Paper

Document Scope

- Intended to be broadly **applicable** to all **technologies**, **platforms**, **programming languages**
- **Not** create **new secure software development practices** nor define **new terminology**, etc.
- Document **a subset of established practices** that should be particularly helpful for the target audiences (organizations with robust secure software development practices in place should not be negatively affected)
- Makes **recommendations without prescribing** exactly how to meet them
  - The most important thing is meeting the **recommendation** and not the mechanism used to do so. For example, one organization might automate a particular step, while another might use manual processes instead.
  - This document will define the **characteristics** the secure software development practices should achieve, such as **consistency, traceability, and repeatability**, and it will encourage **use of tools and automated workflows** instead of manual human labor, but it will not specifically recommend tool acquisition.

Document Scope

- Recommendations are **not** based on an assumption of all organizations having the **same security objectives and priorities**
- Recommendations reflect that each software producer may have unique security assumptions and each software consumer may have **unique security needs**.
  - While the desire is for each security producer to follow all the recommendations, the expectation is that the degree to which each recommendation is implemented will **vary based on the producer's security assumptions**.
  - The recommendations will provide **flexibility for implementers**, but they will also be **clear** to **avoid** having too much open to **interpretation**.
- Focus on the **practices** that will be the most helpful
- Will **not** provide huge lists of **complex things** to do

Proposed Approach

- Adopt an approach similar to the Framework for Improving Critical Infrastructure Cybersecurity, also known as **the NIST Cybersecurity Framework (CSF)**
- Provide **a common language** to describe **fundamental, sound secure software development practices** based on **established standards, guidance,** and **secure software development practice documents**
- Can be **used** by organizations **in any sector or community** regardless of **size** or **cybersecurity sophistication**
- Can be applied to software developed to support information technology **(IT),** industrial control systems **(ICS),** cyber-physical systems **(CPS),** or the Internet of Things **(IoT)**
- **Facilitate the communications** about secure software practices amongst both **internal and external organizational stakeholders**
  - **Business owners**, **software developers**, and **cybersecurity professionals** within an organization
  - **Software consumers** who want to define required or **desired characteristics** for software in their **acquisition processes** in order to have higher-quality software (particularly with fewer security vulnerabilities)
  - **Software producers** who want to express their **secure software practices** to their customers or to define **requirements for their suppliers**

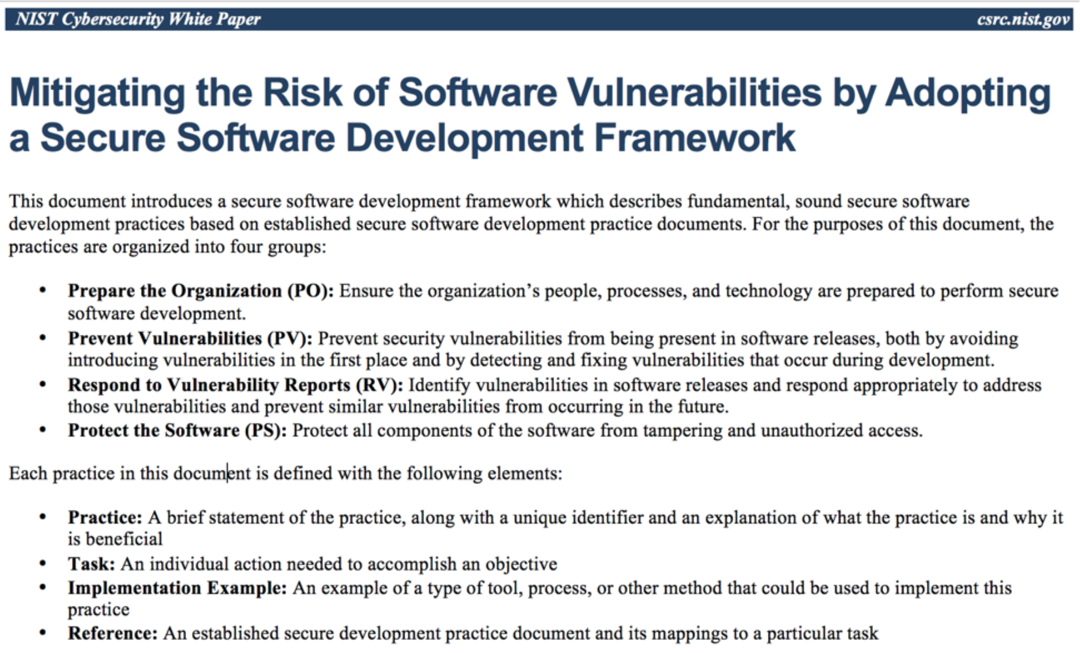
Secure Software Development Framework

Secure Software Development Practices

- The **secure software development practices**
  - are **not a comprehensive** set; instead, they are particularly important items
  - integrate into the **existing software development workflow and automated toolchain**
  - are applicable to **any software development lifecycle (SDLC) model**, **standard**, or **methodology**
  - can be applied to **any programming language** or **development environment** and **any operating environment**
- Expertise in secure software development not required to understand the practices

Current State

- NIST cybersecurity white paper (internal draft)
- Socialize it with industry partners
- Gather initial feedback about the current approach
- Release public draft in May 2019



NIST Cybersecurity White Paper *csrc.nist.gov*

Mitigating the Risk of Software Vulnerabilities by Adopting a Secure Software Development Framework

This document introduces a secure software development framework which describes fundamental, sound secure software development practices based on established secure software development practice documents. For the purposes of this document, the practices are organized into four groups:

- **Prepare the Organization (PO):** Ensure the organization's people, processes, and technology are prepared to perform secure software development.
- **Prevent Vulnerabilities (PV):** Prevent security vulnerabilities from being present in software releases, both by avoiding introducing vulnerabilities in the first place and by detecting and fixing vulnerabilities that occur during development.
- **Respond to Vulnerability Reports (RV):** Identify vulnerabilities in software releases and respond appropriately to address those vulnerabilities and prevent similar vulnerabilities from occurring in the future.
- **Protect the Software (PS):** Protect all components of the software from tampering and unauthorized access.

Each practice in this document is defined with the following elements:

- **Practice:** A brief statement of the practice, along with a unique identifier and an explanation of what the practice is and why it is beneficial
- **Task:** An individual action needed to accomplish an objective
- **Implementation Example:** An example of a type of tool, process, or other method that could be used to implement this practice
- **Reference:** An established secure development practice document and its mappings to a particular task