



Executing on

NIST SP 800-190



Contents

- 3 NIST Special Publications**
- 4 Challenges**
- 5 Twistlock**
- 6 Securing the Stack**
- 6 Vulnerability Management**
- 8 Compliance**
- 10 Runtime Analysis**
- 11 Malware**
- 11 Embedded Secrets**
- 12 Image Trust**
- 12 Conclusion**
- 13 Appendix: Mapping Twistlock to NIST SP 800-190**

Executing on NIST SP 800-190

The National Institute of Standards and Technology (NIST) recently released Special Publication (SP) 800-190, which provides guidance on securing application containers and related ecosystem components.

NIST SP 800-190 thoroughly describes the security risks and associated countermeasures for safeguarding containerized apps. Organizations, however, are left to fend for themselves on the tactical front. If you're securing containerized apps, then you'll likely want to deploy the controls it describes. Open source offers a few disparate solutions, while traditional commercial offerings still struggle to address the concerns unique to containers.

Twistlock is a purpose-built software suite for securing cloud-native workloads. Since our inception, we've been developing technology to address the types of risks described in NIST SP 800-190.

NIST Special Publications

NIST develops standards and guidelines for all federal computer systems (except classified computer systems, which fall under the jurisdiction of the National Security Agency). NIST produces the 800 series of Special Publications. These publications draw from industry, government, and academia to provide computer, cyber, and information security guidelines and recommendations.

The impact of SP 800-190 is twofold:

- For government agencies, the impact is immediate. Government agencies are expected to comply with NIST security standards and guidelines within a year of a publication's release date. Information systems that are currently under development are expected to be compliant upon deployment. Government agencies need to grok the guidelines in SP 800-190 now, and plan how to roll them out. Twistlock implements most of the countermeasures in SP 800-190 and can help immediately.
- For industry, there's more time. As companies plan their rollout, they can look to this publication for best practices. NIST special publications are useful to all security practitioners and they are available for free. To date, there are still very few good resources on securing container environments. Other notable examples are the Center for Internet Security (CIS) Docker and Kubernetes Benchmarks.

Challenges

The methods for securing containers have morphed alongside the evolution of infrastructure. Docker streamlines how you package, store, and deploy apps. Because containers encapsulate all their dependencies, they can move easily from a development environment, to a test environment, to a production environment, making frequent deployments by way of automation the new reality.

Apps built using the microservices architecture consist of many interrelated entities, and the number of deployed entities can mushroom as orchestrators seamlessly scale your apps up (or down) according to demand.

Scale

Google launches 2 billion containers a week in their infrastructure. Organizations have to think about the controls required to secure such massive deployments. The old method of manually creating and maintaining security rules for each entity is impractical.

Rate of change

Deployment cycles have been chopped down from one release per quarter to several releases per week. With each release, apps can change in subtle ways, and the security rules that protect them must keep up. Rotting rules open cracks in your armor that attackers can exploit.

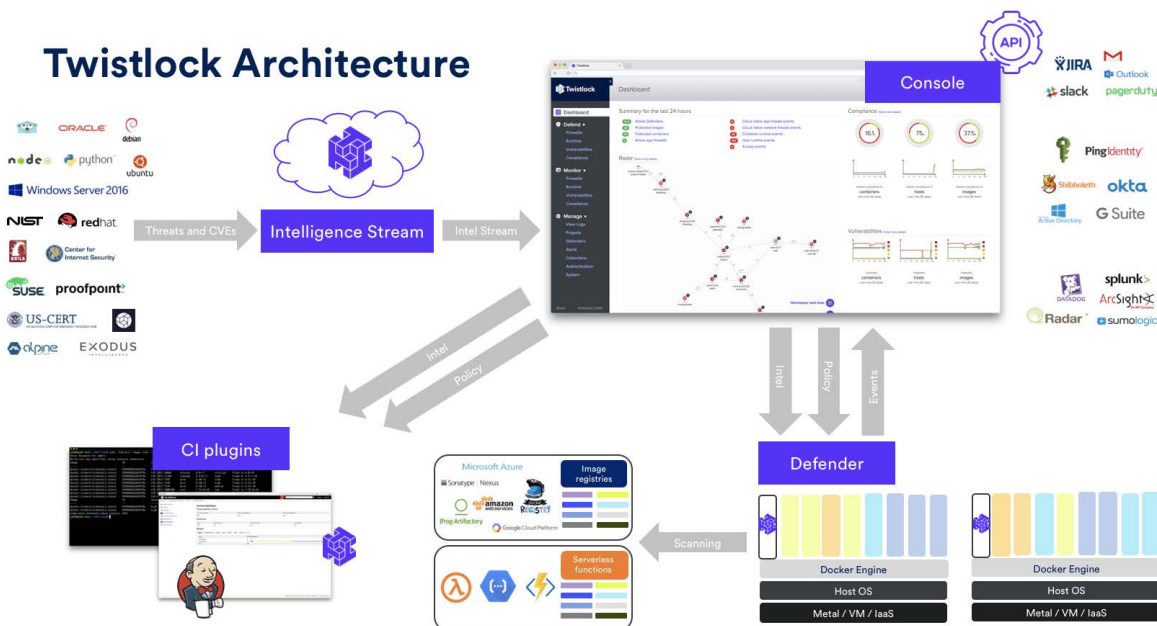
Lifecycle

Security is no longer just the domain of a single team or person. It has to be weaved into the automated processes that build and deploy our software. “Quality” gates let security teams inject policies at the right points in the pipeline. The gates are frictionless when there aren’t issues, but they stall the pipeline when critical issues must be addressed.

Twistlock

Twistlock is the leading provider of container and cloud native cybersecurity solutions. From precise, actionable vulnerability management tools to automatically deployed runtime protection and firewalls, Twistlock protects applications in all stages of their life cycle. Purpose built for containers, serverless, and other leading technologies, Twistlock gives developers the speed they want, and CISOs the controls they need.

A container environment, in general, encompasses your images, containers, hosts, Docker daemons, registries, and orchestrator. The following diagram provides an overview of the major components in a Twistlock deployment:



Twistlock Intelligence Stream

Live feed that delivers the latest threat data to your Twistlock installation.

Console

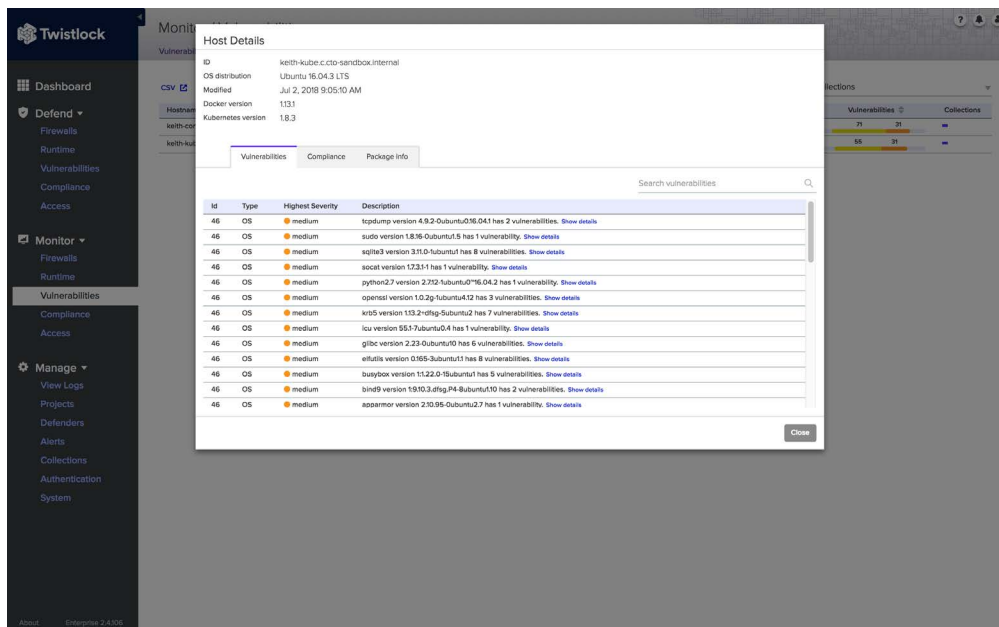
Management interface for declaring security policies and monitoring the health of a deployment. Provides an API for controlling the product programmatically.

Defender

Security agent installed on any machine that runs containers. Enforces your policies.

Securing the Stack

Securing a container environment requires securing the rest of the underlying stack. Organizations should regularly check for and apply updates to all software components installed on the host OS. Cloud-native operating systems, such as CoreOS, ship with auto-updating enabled by default. Twistlock adds another layer of defense by continually scanning the host OS for vulnerable packages that could be entry points for an attack. The following screenshot shows the scan report for a host in the container environment:



The screenshot displays the Twistlock interface with a 'Host Details' window open. The window shows the following information:

- ID: k8th-kube.c-cto-sandbox-internal
- OS distribution: Ubuntu 18.04.3 LTS
- Modified: Jul 2, 2018 9:05:10 AM
- Docker version: 1.13.1
- Kubernetes version: 1.8.3

Below this information is a table of vulnerabilities:

Id	Type	Highest Severity	Description
46	OS	medium	topgun version 4.8.2-0ubuntu0~16.04.1 has 2 vulnerabilities. Show details
46	OS	medium	sofs version 1.8.16-0ubuntu1.5 has 1 vulnerability. Show details
46	OS	medium	apfs3 version 311.0-0ubuntu1 has 8 vulnerabilities. Show details
46	OS	medium	soat version 1.23.1-1 has 1 vulnerability. Show details
46	OS	medium	python27 version 2.7.12-0ubuntu0~16.04.2 has 1 vulnerability. Show details
46	OS	medium	openat version 1.0.2p-0ubuntu4.12 has 3 vulnerabilities. Show details
46	OS	medium	krb5 version 1.13.2+dfsg-0ubuntu2 has 7 vulnerabilities. Show details
46	OS	medium	lca version 55.1-7ubuntu0.4 has 1 vulnerability. Show details
46	OS	medium	glibc version 2.23-0ubuntu1 has 6 vulnerabilities. Show details
46	OS	medium	elfutils version 0.155-3ubuntu1 has 8 vulnerabilities. Show details
46	OS	medium	busybox version 1.12.2-0-15ubuntu1 has 5 vulnerabilities. Show details
46	OS	medium	bind9 version 1:9.10.3.dfsg.P4-8ubuntu1.10 has 2 vulnerabilities. Show details
46	OS	medium	apparmor version 2.10.95-0ubuntu2.7 has 1 vulnerability. Show details

You should ensure all authentication to the OS is audited, anomalies are monitored, and any escalation to perform privileged operations is logged. Twistlock catches and logs all sudo and sshd events on any host protected by Defender. They are reported when someone establishes an SSH connection to a host and runs commands with elevated privileges.

Vulnerability Management

Organizations should employ container-specific vulnerability management tools and processes to look for Common Vulnerabilities and Exposures (CVEs) across all phases of the container lifecycle. You should upgrade any instances at risk, and ensure that orchestrators only permit deployments of properly maintained runtimes.

With containers, engineers now define the environment in which their apps run, and their decisions directly impact security of the environment. Organizations need to arm their engineers with the right tools to find and fix vulnerabilities. Twistlock

provides a Jenkins plugin that lets you incorporate vulnerability scanning in the continuous integration/continuous deployment (CI/CD) pipeline. This plugin helps developers find and fix security defects before images ever get to the registry. The plugin also serves as a communications conduit between security and development teams. Twistlock lets the security teams create policies and standards to which the engineering teams must build their images. Non-compliant images fail the build. For example, a security team could define a policy that fails the build if any critical vulnerabilities are found.

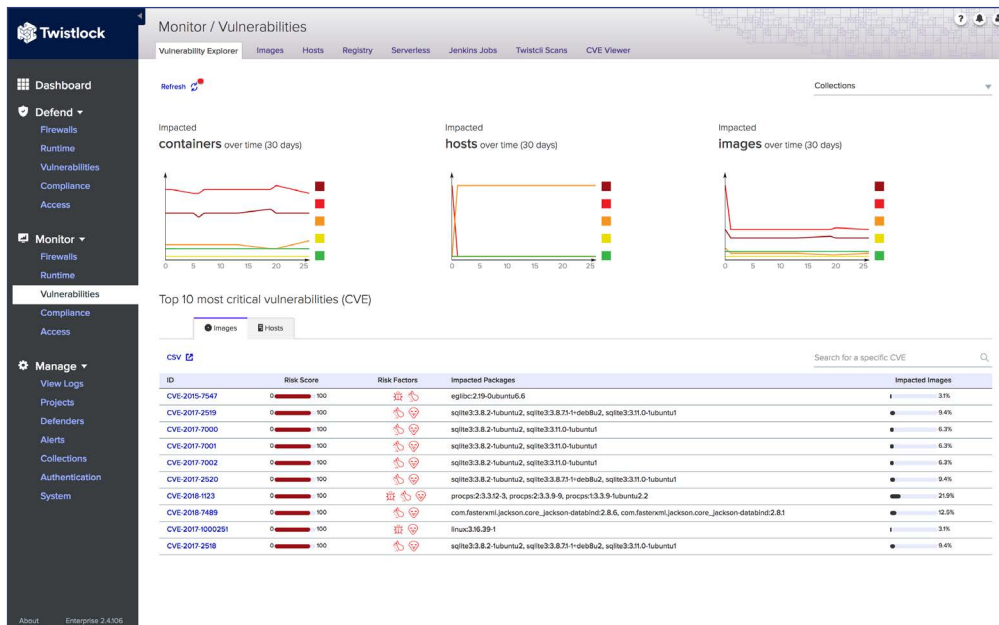
Besides the Jenkins plugin, Twistlock’s command line utility, `twistcli`, offers a number of capabilities for the individual developer. For one, they can scan images for vulnerabilities and compliance issues before checking code into a source control system. Because `twistcli` runs from the command line, users can integrate Twistlock scanning capabilities into custom tooling.

Drift is another issue. After engineers build a clean image, they push it to registry for distribution. An image deemed clean today, may not be clean tomorrow. Twistlock continually monitors the images in your registries and container environment for vulnerabilities. The Twistlock Intelligence Stream (a threat feed) is updated several times a day with new threat data, and that threat data is assessed against all containers in the environment. Scan reports provide intel about which images are vulnerable, and where they reside.

The following screenshot shows a scan report. Twistlock scans images and containers for CVEs, malware, compliance issues, and zero-day vulnerabilities.

Registry	Repository	Tag	Hosts	Vulnerabilities	Risk Factors	Running	Collections
docker.io	moreio/docker-whale	latest	keith-console	167	171	12	true
docker.io	moreio/httpd	latest	keith-console	63	144	112	4
docker.io	library/wordpress	latest	keith-console	95	93	25	25
docker.io	library/rabbitmq	3.6.8	keith-kube	17	39	17	17
docker.io	weavek8sdemos/user-db	0.4.0	keith-kube	14	40	17	17
docker.io	weavek8sdemos/catalogue-db	0.3.0	keith-kube	12	26	17	17
docker.io	library/mysql	latest	keith-console	14	30	15	15
gcr.io	google_containers/kube-proxy-amd64	v1.8.3	keith-kube	10	27	14	14
docker.io	library/mongo	latest	keith-kube	33	37	14	14
docker.io	weavek8sdemos/front-end	0.3.12	keith-kube	8	4	2	2
docker.io	weavek8sdemos/orders	0.4.7	keith-kube	1	0	0	0
docker.io	weavek8sdemos/queue-master	0.3.1	keith-kube	1	0	0	0
docker.io	weavek8sdemos/shipping	0.4.8	keith-kube	1	0	0	0
docker.io	weavek8sdemos/carts	0.4.8	keith-kube	1	0	0	0
docker.io	moreio/metools	latest	keith-console	8	5	8	8
quay.io	coreos/etcd	v3.1.10	keith-kube	1	4	2	2
docker.io	weavek8sdemos/payment	0.4.3	keith-kube	3	2	2	2
docker.io	weavek8sdemos/...	0.4.4	keith-kube	3	2	2	2
docker.io	weavek8sdemos/catalogue	0.3.5	keith-kube	3	2	2	2
gcr.io	google_containers/kube-apiserver-amd64	v1.8.3	keith-kube	2	1	1	1

Policies let you take action on intel from the scan reports. You can create policies that specify how to handle vulnerable images. For example, a policy might block the deployment of a container to the production environment if it has a critical severity vulnerability.



Finally, organizations need a mechanism to oversee their container environment. Vulnerability Explorer provides a number of tools:

- A ranked list of the most critical vulnerabilities in your environment, which can be used to prioritize your remediation efforts. The ranking is based on a risk score that takes into account the characteristics of the CVE and the context with which it appears in your environment.
- Risk trees for individual vulnerabilities. Risk trees show you how and where you are exposed to a given vulnerability. For example, your CISO might want to understand your exposure to CVE-2018-1234. Vulnerability Explorer can show you exactly which running containers are affected by CVE-2018-1234, which images they're derived from, and on which hosts they reside.

Compliance

Like many software packages, out-of-the-box Docker has been configured for convenience. Organizations should adopt tools and processes to validate and enforce compliance with secure configuration best practices for container images and runtimes. For best results, these compliance checks should be automated.

The Center for Internet Security (CIS) publishes a document called the Docker Benchmark that defines the security best practices for building Docker images, running containers, and configuring your hosts. Twistlock adapts the Docker Benchmark recommendations into discrete checks. We've also adapted the Kubernetes Benchmark and General Linux Benchmark into discrete checks.

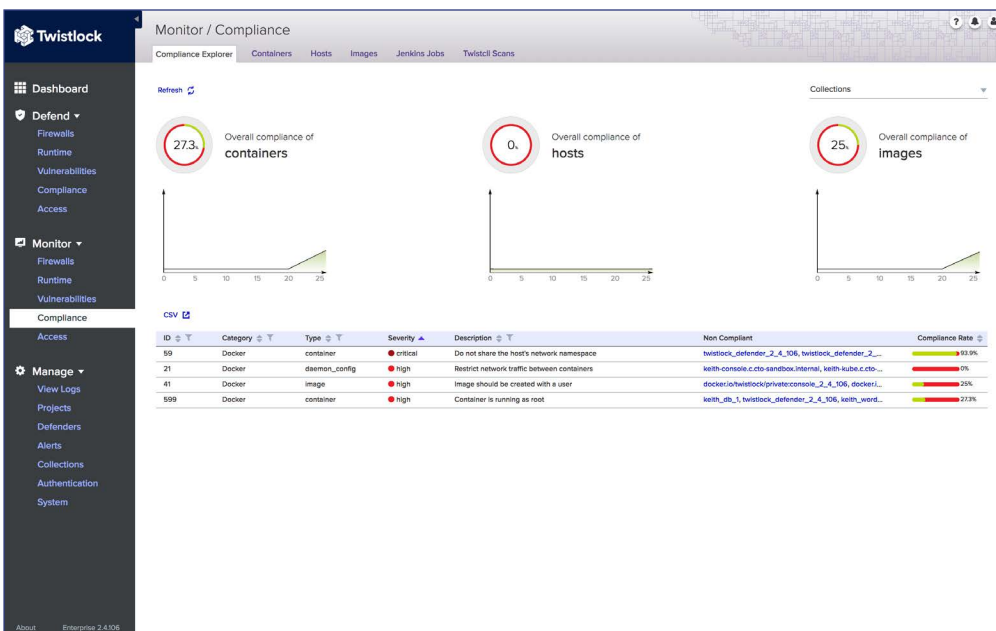
The discrete checks can be enabled to enforce and maintain compliance to specific requirements from industry and government standards, such as PCI DSS and HIPAA. For example, the Docker Benchmark check 4.1 validates that an image is configured to run as a non-root user. All security standards prescribe least privilege access, and running almost any container as root (which is the default setting) almost certainly yields excessive privileges.

Like vulnerability management, Twistlock lets you set policies around compliance checks. Reports on compliance aren't enough though, so Twistlock lets you protect critical domains by preventing the deployment of non-compliant images.

Twistlock has graded every check in the benchmarks we support. By default, all critical and high severity checks are set to alert. This grading system immediately surfaces the most serious issues in your environment. Of course, you can further tailor the list of enabled checks to enforce compliance to a specific industry or government standard. The following screenshot shows Compliance Explorer, which summarizes your environment's compliance to the enabled checks. Compliance officers and auditors can use this dashboard to quickly validate compliance to a predefined checklist.

Besides the out-of-the-box CIS Docker Benchmark checks, Twistlock also supports two methods of implementing your own custom checks:

- Sandboxed bash scripts.
- Extensible Checklist Configuration and Description Format (XCCDF)



Runtime Analysis

Organizations should leverage machine learning to automatically profile containerized apps and build protection profiles for them.

Security tools tend to be reactive, and can end up as relics when new threats emerge. You cannot depend on just static identifiers, such as CVEs and malware hashes, to secure your environment. Twistlock runtime defense secures your containers from emerging threats by modeling the intent of each image in your environment and then using those models to detect abnormal activity.

Twistlock builds predictive models for each image in your environment. Models comprise of rules that whitelist specific activities and, conversely, blacklist everything else.

Rules in the whitelist models span four dimensions:

- **Process control:** Identify a list approved processes that can be started and run in a container. Invalid or unexpected process execution raises an exception.
- **Networking:** Identify ports that should be open and monitor for traffic to malicious endpoints. Creating unexpected network listeners or connecting to known malicious network destinations raises an exception.
- **File system:** Detect changes to the file system. Changes to binaries or protected configuration files, or writes to unexpected locations, raises an exception. Binaries downloaded into containers are assessed against MD5 checksums for known malicious software.
- **System calls:** Profile the container according to the system calls it makes.

After a model is activated, sensors continually monitor running containers on these four dimensions. Policies can be created to specify how anomalies should be handled. For example, if an attacker breaches a container and tries to download malware with curl, the connection would be flagged as an anomaly, and Twistlock can alert you, kill the anomalous curl process, or stop the entire container. Additional whitelist or blacklist rules can be created to complement models that don't fully capture the range of known good activities in a container.

Nothing is required to activate runtime analysis. Models are automatically generated the first time an image is seen in the environment, and Twistlock automatically enforces the rules in the model. Automatic modeling helps in two ways:

- It lets you scale your security with apps. No additional manpower is required to create new security rules for new containers.
- It keeps your security rules tight as your app evolves. Twistlock automatically profiles and creates new models for updated images.

Malware

Organizations should adopt tools and processes to monitor images for malware, both at rest and runtime.

Twistlock scans all images in your environment for malware. Twistlock runtime defense monitors running containers for malware being downloaded into the container file system. Hashes for known malware is delivered to your environment through the Twistlock Intelligence Stream. You can augment data from Twistlock's feed with your own custom data.

Embedded Secrets

Organizations should never store sensitive data in image files.

Twistlock provides a control that detects secrets embedded in images. You can then create policies that take action by alerting or actively blocking the deployment of images that contain embedded secrets. This compliance check forces developers to utilize safer methods for using secrets.

Twistlock detects:

- Private keys in the container file system.
- Environment variables that expose sensitive data.

Twistlock integrates with a number of secrets stores. After integrating with a secrets store, you can set up policies for injecting secrets into specific containers. The following secrets stores are supported:

- HashiCorp Vault
- CyberArk Enterprise Password Vault
- Azure Key Vault
- AWS Secretes Manager
- AWS System Manager Parameter Store

Image trust

Organizations should designate images and registries that have been fully vetted as trusted, then ensure that only these images are allowed to run in your environment.

Twistlock lets you define an explicit list of trusted repositories and/or images, and then create rules that only allows only those images to run in your environment. Twistlock's image trust feature reduces the risk of running sabotaged or unauthorized images in your environment (SP 800-190 countermeasure 4.4.5).

For example, you could create a policy that limits the production environment to running just approved images from your private (trusted) registry, while blocking anything from external public repositories, such as Docker Hub.

Conclusion

There is good reason that there is tight alignment between the recommendations in SP 800-190 and Twistlock's offering. Since our inception, we've been thinking about the right way to solve the problems related to container security. We're not repurposing old tools to solve new problems. We're building cloud-native tools to address the challenges and opportunities unique to containers.

NIST SP 800-190 contains the current best thinking on securing containers, and Twistlock implements most of the countermeasures described in it. When apps scale out, the old method of manually creating and maintaining security rules becomes impractical. Twistlock is purpose-built for securing container environments at scale. Automation plays a key role. Runtime analysis and machine learning automatically create and enforce policies to secure container workloads across the environment.

Appendix: Mapping Twistlock to NIST SP 800-190

This section maps the recommended countermeasures in NIST SP 800-190 to the controls and capabilities offered by Twistlock. Twistlock provides comprehensive coverage for nearly all countermeasures.

4.1 Image Countermeasures

4.1.1 Image Vulnerabilities

Employ container-specific vulnerability management tools and processes.

The Twistlock scanner creates a manifest of components in each image in your environment, then uses these manifests to quickly evaluate new threat data against your images, and report on them in near real-time

Twistlock plugs into all phases of the container lifecycle

CI/CD pipeline: Twistlock's native Jenkins plugin or stand-alone CLI utility can scan images for vulnerabilities and compliance issues in a post-build step. Use centrally defined policies to pass or fail builds

Registry: Twistlock scans your registries at a configurable interval (default 24h) to ensure images are maintained and updated as new threat data becomes available and configuration requirements change

Images and running containers on hosts: Twistlock scans the images and containers on your hosts at a configurable interval (default 24h) to ensure that they are maintained and updated as new threat data becomes available and configuration requirements change

Dev: Twistlock can create JIRA issues (or send emails or Slack messages) when new critical vulnerabilities are discovered. JIRA issues close the loop by ensuring critical issues are submitted for fixes directly in engineering's planning tool

Twistlock provides comprehensive Common Vulnerabilities and Exposures (CVE) data for all popular base layers:

- Alpine
- Amazon Linux container image
- BusyBox
- CentOS
- Debian
- Red Hat Enterprise Linux
- SUSE
- Ubuntu
- Windows Server

Twistlock provides comprehensive CVE data for all major app frameworks:

- Ruby
- Java
- Python
- Nodejs

4.1.1 Image Vulnerabilities

Employ container-specific vulnerability management tools and processes.

Twistlock Vulnerability Explorer centrally reports all vulnerabilities in your environment and includes the following tools:

Top 10: Ranked list of the most critical vulnerabilities in your environment. Use it to prioritize your remediation efforts

Risk trees: Show how and where you are exposed to a given vulnerability. Example: CVE-2018-1234 can be found in these running containers, derived from these images, which reside on these hosts

Twistlock policies work as quality gates to ensure vulnerable images are detected, and optionally blocked, at all entry points into your environment

CI/CD pipeline: Prevent vulnerable images from being promoted to your registry based on policies that pass or fail builds. Rules let you selectively or globally whitelist CVEs that your security team have classified as benign so that the build pipeline isn't stalled

Registry to Production: Twistlock assesses image contents against your policy before permitting deployment

Public Internet to Production: Twistlock lets you define trusted images (based on origin, base layer, or image ID) and prevents untrusted images from entering sensitive environments

4.1.2 Image configuration defects

Adopt tools and processes to validate and enforce compliance with secure configuration best practices for container images.

Twistlock supports the CIS Docker Benchmark, which, among other things, defines the best practices for building images

Twistlock Labs grades each check (Critical, High, Medium, and Low), and enables the Critical and High checks by default so that you can focus on the most critical issues

Images are periodically rescanned for configuration defects at a configurable interval (default 24h)

Twistlock Compliance Explorer provides a quick view for compliance officers to see the percentage of total checks passed out of the total checks enabled, with a ranked list of the most severe issues

You can write your own compliance checks using simple bash scripts

You can write your own compliance checks using OpenSCAP

Twistlock lets you write policies that alert or block the deployment of non-compliant images

Twistlock lets you define trusted images by base layer(s), and then write policies that alert or block non-compliant images from being deployed

4.1.3 Embedded malware

Adopt tools and processes to monitor images for malware, both at rest and runtime.

Twistlock periodically scans images at a configurable interval (default 24h) for malware during all phases of the image lifecycle: build (Jenkins plugin or twistcli), registry, and deployment on the hosts where they've been pulled

Twistlock runtime protection (for running containers) automatically detects when any malicious binary written to a container file system. It also detects any changes to any binaries or certs anywhere in a container

Intelligence Stream is updated with new threat data daily, and images are automatically re-assessed against the new data

You can augment the Twistlock feed with your own malware signatures

Twistlock Labs continually works on advanced protection for our customers: We've implemented a check for crypto miners, which can enter your environment as a Trojan, a build time dependency (e.g. base layer), etc.

4.1.4 Embedded clear text secrets

Never store sensitive data in image files.

Twistlock provides out-of-the-box compliance checks that detect easy-to-steal secrets:

- Private keys in container images
- Unencrypted secrets in env vars

Twistlock integrates with popular enterprise secret management systems:

- HashiCorp Vault
- CyberArk Enterprise Password Vault
- Azure Key Vault
- AWS Secretes Manager
- AWS System Manager Parameter Store

Twistlock lets you restrict access to secrets with policies that specify which secrets get injected into which containers

4.1.5 Use of untrusted images

Designate images and registries that have been fully vetted as trusted, then ensure that only these images are allowed to run in your environment.

Twistlock lets you centrally define which images are trusted, and then define enforcement policies that block the deployment of anything other than trusted images.

Trusted images can be defined by:

- Origin (registries)
- Image ID
- Base layer(s)

Policies offer rich filters that let you ratchet up restrictions in certain envs (e.g. prod) but loosen them in others (e.g. dev).

You can configure Twistlock to scan your registries to report issues as new vulnerabilities are discovered and compliance policies evolve.

4.2 Registry Countermeasures

4.2.1 Insecure connections to registries

Only connect to registries over encrypted channels.

Twistlock policies can block images from being pulled from non-approved locations. This policy forces you to access only the registries that support encrypted transfer

4.2.2 Stale images in registries

Implement processes to ensure old vulnerable images are never deployed.

Twistlock continually scans, re-assesses, and reports on vulnerabilities and compliance issues in images stored in registries

Twistlock Intelligence Stream is updated daily with new threat data, and images are re-assessed against the new data in near real-time

Policies can alert, and optionally block, the deployment of registry images with critical vulns and/or compliance issues

Twistlock API and the twistcli command line utility can be integrated into a larger, custom framework that automatically scans and prunes vulnerable images from a registry

4.2.3 Insufficient authentication and authorization restrictions

Require authentication to ensure that only images from trusted entities can be added.

Twistlock role-based access control policies can restrict push and pull access to private registries

You can integrate Twistlock with your org's directory service or identity provider (Active Directory, OpenLDAP, SAML), then define rules that restrict who can pull and push to your registry

The Twistlock scanner can be integrated into your build pipeline, so that images can only be pushed to your registry after they pass a vulnerability scan and compliance assessment

4.3 Orchestrator Countermeasures

4.3.1 Unbounded administrative access

Orchestrators should use a least privilege access model.

Twistlock Access Control ships with a default deny-all access control rule for Docker and Kubernetes commands. Any permitted activity must be explicitly whitelisted

4.3.2 Unauthorized access

Access to cluster-wide administrative accounts should be tightly controlled.

Twistlock supports integration with your organization's directory service or identity provider (Active Directory, OpenLDAP, SAML). After integration, you can grant access to orchestrator commands on a user-by-user or group-by-group basis.

You can define access control rules based on filters and pattern matching expressions for host names, image names, container names, and/or labels.

All accesses are logged, including commands that are permitted and commands that violate policy.

Policies can be configured to raise alerts or block the command entirely.

Twistlock supports multi-factor authentication built on x.509 certificates, such as smart cards.

4.3.3 Poorly separated inter-container network traffic

Separate network traffic into discrete virtual networks.

Twistlock Cloud Native Network Firewall (CNNF) operates as an east-west firewall between containers, limiting damage by preventing attackers from moving laterally through your environment when they have already compromised one part of it

CNNF automatically learns the valid connections between containers, and blocks all other attempted connections

4.3.4 Mixing of workload sensitivity levels

Configure orchestrators to isolate deployments to specific sets of hosts by sensitivity levels.

Twistlock supports various deployment models:

For strict isolation, where you deploy one cluster per sensitivity level, Twistlock can secure each environment separately. An admin deploys Twistlock to the segregated environment, centrally defines which users and groups have access to it, and then hands it off to the responsible team to manage the rules, policies, and settings required to secure the environment

For shared clusters, Twistlock can leverage your labeling and naming schemes to:

Selectively enforce security policies. Twistlock rules let you specify attributes, such as image name, container name, host name, and labels, to selectively target specific resources in an environment

Group related resources (images, containers, hosts, labels) for easier management and visualization in the dashboard

Append your Docker or Kubernetes labels to Twistlock events. For example, if the label “cost-center” is applied to all your resources, the you can direct Twistlock to append the cost-center key and value to any Twistlock event triggered by a resource with the cost-center label

4.3.5 Orchestrator node trust

Twistlock scans the underlying host for vulnerabilities

Twistlock scans the underlying host for compliance issues from the following industry standard benchmarks:

- CIS Docker Benchmark
- CIS Kubernetes Benchmark
- CIS General Linux Benchmark (to secure the host OS)

Twistlock host runtime protection extends our model-based approach to secure systemd services running on your hosts

Twistlock API and the twistcli command-line tool can be integrated into larger, custom framework that expels nodes from the cluster when they fall out of compliance

4.4 Container Countermeasures

4.4.1 Vulnerabilities within the runtime software

Carefully monitor running containers for vulnerabilities. When problems are detected, quickly remediated them.

Running containers are continually re-assessed for vulnerabilities at a configurable scan interval (default 24h)

Any newly created container is immediately assessed for vulnerabilities before it is instantiated. If it violates your policy, Twistlock raises an alert and optionally blocks it from being deployed

As part of the alert machinery, Twistlock can be configured to open JIRA issues for new critical vulnerabilities to close the loop and ensure vulns are scheduled for timely remediation. Email and Slack alerts are also supported.

Twistlock risk trees lists all images, containers, and hosts that are vulnerable to a specific CVE. Risk trees are useful because they show you how and where you are exposed to a given vulnerability. For any given vulnerability, you can see a map of all affected containers and images, and on the nodes where they reside.

4.4.2 Unbounded network access from containers

Control the egress network traffic sent by containers

Twistlock Runtime Defense builds predictive models for each image in your environment. Network models identify when attackers:

- Create unexpected network listeners (ingress)
- Connect to unexpected network destinations (egress)
- Connect to known malicious destinations (egress)

Twistlock Cloud Native Network Firewall (CNNF) uses machine learning to model network traffic between containers, then automatically creates rules that whitelist known good traffic

CNNF works as an east-west firewall between containers, limiting damage by preventing attackers from moving laterally through your environment when they have already compromised one part of it (egress)

4.4.3 Insecure container runtime configurations

Automate compliance with container runtime configuration standards.

CIS Docker Benchmark

- Twistlock implements the CIS Docker Benchmark
- Twistlock continuously assesses your configuration settings across the environment and actively enforces them according to your policies
- Twistlock Labs graded all checks (Critical, High, Medium, and Low). Critical and High checks are enabled by default so you can focus on the most serious issues in your environment
- Just by deploying Twistlock to your environment (no configuration required), Twistlock can alert you to critical compliance issues

seccomp

- Twistlock Labs provides curated custom seccomp profiles for widely used containerized apps, such as Apache and MongoDB. These custom seccomp profiles are injected into the relevant containers when they're started
- You can further refine injected seccomp profiles with policies that whitelist or blacklist specific system calls

4.4.4 App vulnerabilities

Implement container-specific intrusion detection systems.

Twistlock Runtime Defense secures your containers by automatically modeling the intent of each image in your environment and then using those models to detect abnormal activity

Nothing is required to activate Twistlock Runtime Defense; protection is enabled by default without any extra configuration required

Models have four dimensions: processes, networking, file system, and system calls. Runtime Defense sensors can protect against all of the events outlined in this requirement:

- Invalid or unexpected process execution
- Invalid or unexpected system calls
- Changes to protected configuration files and binaries
- Writes to unexpected locations and file types
- Creation of unexpected network listeners
- Traffic sent to unexpected network destinations
- Malware storage or execution

Twistlock ships with CIS Docker Benchmark compliance check 5.12

automatically enabled: Raise an alert if a container mounts its root filesystem with any permission other than read-only

4.4.5 Rogue containers

Twistlock offers several controls to address rogue containers:

Role based access control lets you set policies for who can access which resources in which environments with which commands

Rules let you target specific hosts, images, container, and labels with pattern matching expressions

Twistlock logs all Docker and Kubernetes commands, along with the identity of the user that ran them

Before any container is started, Twistlock assesses its vulnerabilities and compliance issues against your policies. If it violates your policy, an alert is raised, and the container is optionally blocked from running

4.5 Host Countermeasures

4.5.1 Large attack surface

Twistlock explicitly supports container-optimized OSs, such as CoreOS and Google's Container-Optimized OS

Twistlock supports the CIS General Linux Benchmark, which defines the best practices for securely configuring a Linux host

- Twistlock Labs has graded all checks, with Critical and High checks automatically enabled to prioritize the most serious issues

Twistlock Runtime Defense for hosts extends our model-based approach to secure systemd services running on your hosts

4.5.3 Host OS component vulnerabilities

Twistlock scans the host OS for vulnerabilities and malware and at a configurable interval (default: 24h)

Twistlock Intelligence Stream contains vulnerability data for orchestrator infrastructure components

4.5.4 Improper user access rights

Twistlock logs all ssh sessions

Twistlock logs all commands run in an interactive session

Twistlock logs all sudo commands in an interactive session

Twistlock logs all Docker commands that induce a state change (e.g. docker create, docker run, docker commit) and, optionally, all read-only commands (e.g. docker ps, docker images)

Twistlock can direct all audit events to syslog in RFC5424-compliant format for integration with monitoring tools, such as Splunk, DataDog, and SumoLogic

4.5.5 Host file system tampering

Twistlock provides a compliance rule (enabled by default) that raises an alert when a container mounts sensitive host system directories

Twistlock runtime models automatically determine where containers should write in the file system, and then enforce those rules

You can create additional runtime rules that blacklist or whitelist writes to specific directories



Twistlock is the leading provider of container and cloud native cybersecurity solutions for the modern enterprise. From precise, actionable vulnerability management to automatically deployed runtime protection and firewalls, Twistlock protects applications across the development lifecycle and into production. Purpose built for containers, serverless, and other leading technologies — Twistlock gives developers the speed they want, and CISOs the control they need.

Follow Twistlock



Twitter



Facebook



LinkedIn