Twistlock Tech News
September 2017

Topics for this edition:

- **Announcing Twistlock 2.2**
- **Cloud Native Network Firewall**
- **Incident Explorer**
- **Runtime defense for container hosts**
- **Native deployment on Swarm**
- **Integrated Slack and JIRA alerting**
- **Compliance monitoring and enforcement for Kubernetes**

Please feel free to share this newsletter with others in your organization, but let us know if you'd like to share beyond that.

Announcing Twistlock 2.2

We just signed off on Twistlock 2.2, the 11th time we've shipped a major release of Twistlock over the past ~2.5 years. A few other fun facts from GitHub: we've worked on more than 5400 issues, built Twistlock more than 600 times, and shipped 223 customer requested features for our >80 paying enterprise customers over that time!

2.2 is focused on advanced threat analytics and prevention, applying machine learning to automate layer 3 firewalling in containerized environments, providing runtime defense down to the host OS, and delivering comprehensive compliance monitoring and enforcement for Kubernetes. Read on to get more details on these and other highlights in the release.

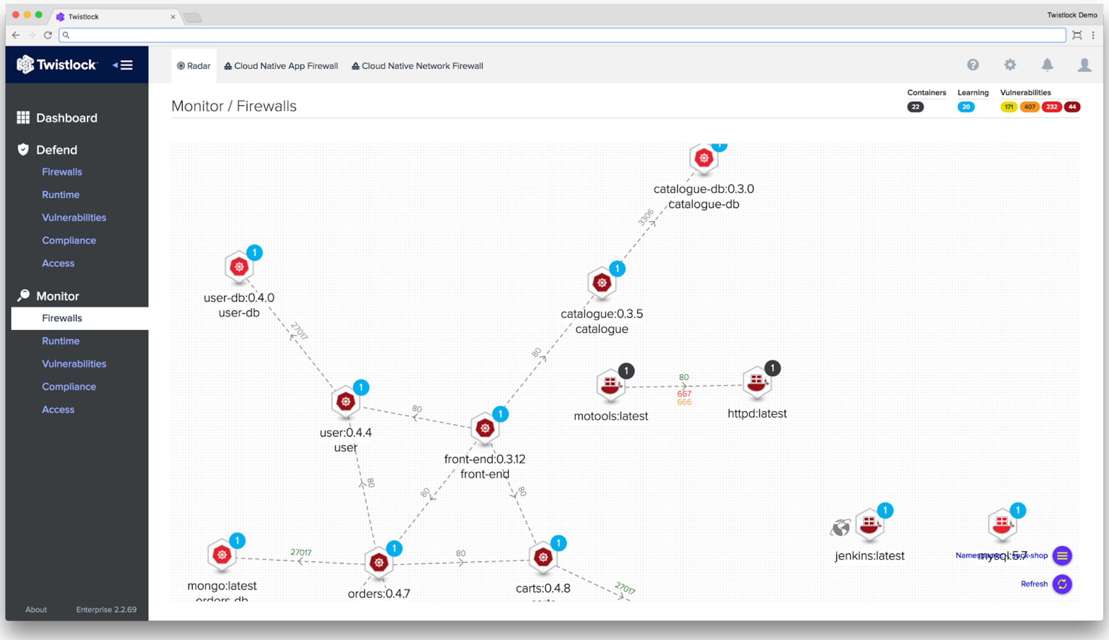
As always, you can download the latest build and view the full release notes at http://support.twistlock.com. Your existing license keys will continue to work normally. Of course, if you have any problems, please just ping us at support@twistlock.com.

-- the Twistlock R&D team
Herzeliya and Baton Rouge

Cloud Native Network Firewall

In 2.1, we shipped a first of it's kind web app firewall designed specifically for containers and cloud native apps. CNAF took the concept of the WAF, endowed it with machine learning driven modeling of the HTTP stream, and enabled 'container aware' configuration. CNAF makes protecting an app as easy as clicking a single button and applying it to any container called my-app*, regardless of what physical host, cloud provider, network, or port that app is running on. With CNNF, we're applying the same intelligence to deliver a container aware, machine learning driven, layer 3 firewall that's more automated and comprehensive than anything else on the market.

The concept of internal network segmentation and compartmentalization is not new. Long considered an important layer of defense in depth, these 'east / west firewalls' limit the scope of damage if one part of your environment is compromised by preventing attackers from moving laterally throughout the rest of it. As organizations adopt containers and cloud native apps, some of the core assumptions traditional firewalls have no longer apply. With containers, traffic is usually encapsulated and encrypted between nodes in an overlay network typically opaque to traditional tools. With containers, the IPs of endpoints are ephemeral and largely irrelevant; no longer can you rely on a manually maintained rule like "from 192.168.1.100 to 192.168.1.200, allow tcp/27017" because you usually don't know or care what IPs a container may be using at any given point in time. Even more critically, though, is that with containers, the total number of endpoints may scale by an order of magnitude, from a few hundred VMs to a few thousand containers. Tools that rely on fragile, manually maintained rules simply can't scale here.



In 2.0, we shipped a layer 3 firewall capability designed to understand and model traffic flows between app components but that relied on Kubernetes CNI plugins for enforcement. While effective, this limited the capability to only Kubernetes environments and required synchronization of policy from Twistlock to the orchestrator. In 2.2, we've refactored this entire solution to work with any orchestrator and to automatically provide enforcement natively within Twistlock. Thus, Twistlock automatically creates models for

every unique version of every app and enforces that only those specific traffic flows are allowed between your microservices. There's no manual interaction required, each time you deploy we learn, model, and enforce automatically without any training of the model or manual rule maintenance needed.

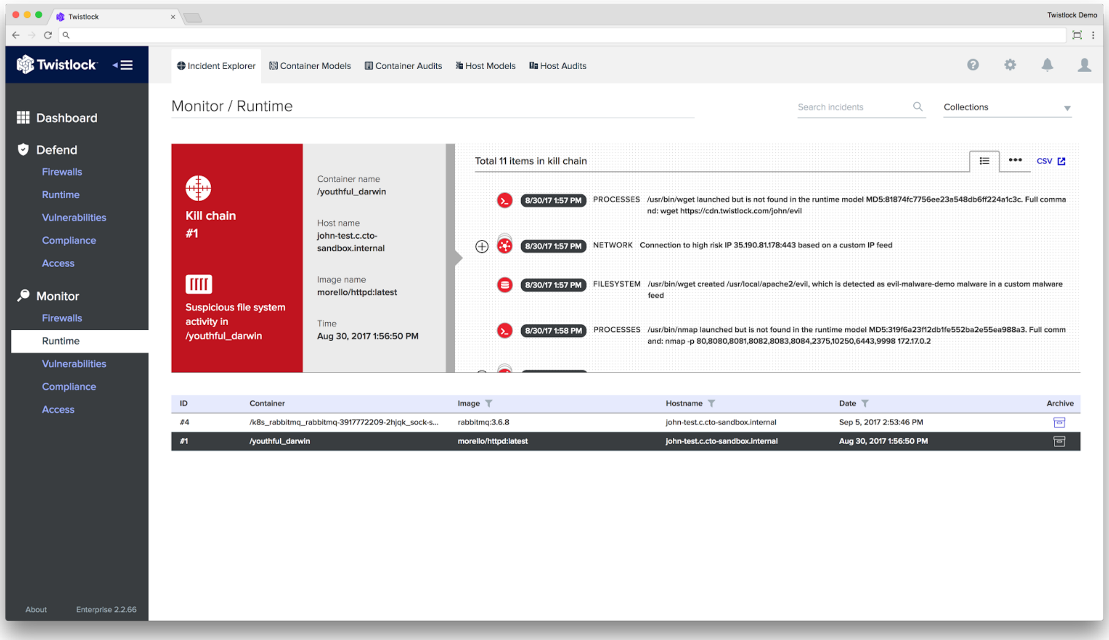
CNNF helps solve a long term security challenge: how to align the needs of apps with a least privilege security model such that only 'good' traffic is allowed to flow. By applying machine learning and automation in a cloud native way, CNNF enables you to centrally model, view, and enforce safe traffic flows across your environment and automatically block anomalies without human involvement.

Read more about Cloud Native Network Firewall.

Incident Explorer

We've often talked about containers helping to improve app security. Because they're minimalistic, declarative, and predictable, you can model everything they should do rather than trying to maintain a list of signatures of every possible bad thing they shouldn't. This machine learning driven approach is at the core of what we've done for runtime since Twistlock 1.0 back in 2015. In 2.2, we're applying the same kind of machine intelligence to identify attack patterns in your environment and displaying them in a clear, human digestible format with a beautiful UI.

Because our models include all the binaries that should run, the file paths accessed, and network ports and flows, we can identify and block anomalous activity. However, prior to 2.2, we recorded and displayed this data as discrete audit events with little direct correlation or analysis. Incident Explorer is a new feature designed to apply our machine intelligence to the correlation and analysis of events that span multiple actions and sensors. For example, if your containerized app is compromised, the attacker is unlikely to just run a single unexpected process and move on. Instead, they may modify configuration files so their compromise persists, establish a new listener to shovel data out of the environment, begin port scanning to map the rest of the environment, and maybe download a rootkit to bond the victim to a command and control node.



Container name
/youthful_darwin

Host name
john-test.c.cto-sandbox.internal

Image name
morello/httpd:latest

Time
Aug 30, 2017 1:56:50 PM

Total 11 items in kill chain

```
8/30/17 1:57 PM  PROCESSES   /usr/bin/wget launched but is not found in the runtime model MD5:81874fc7756ee23a548db6ff224a1c3c. Full command: wget https://cdn.twistlock.com/john/evil
8/30/17 1:57 PM  NETWORK     Connection to high risk IP 35.190.81.178:443 based on a custom IP feed
8/30/17 1:57 PM  FILESYSTEM  /usr/bin/wget created /usr/local/apache2/evil, which is detected as evil-malware-demo malware in a custom malware feed
8/30/17 1:58 PM  PROCESSES   /usr/bin/nmap launched but is not found in the runtime model MD5:319f6a23f12db1fe552ba2e55ea988a3. Full command: nmap -p 80,8080,8081,8082,8083,8084,2375,10250,6443,9998 172.17.0.2
```

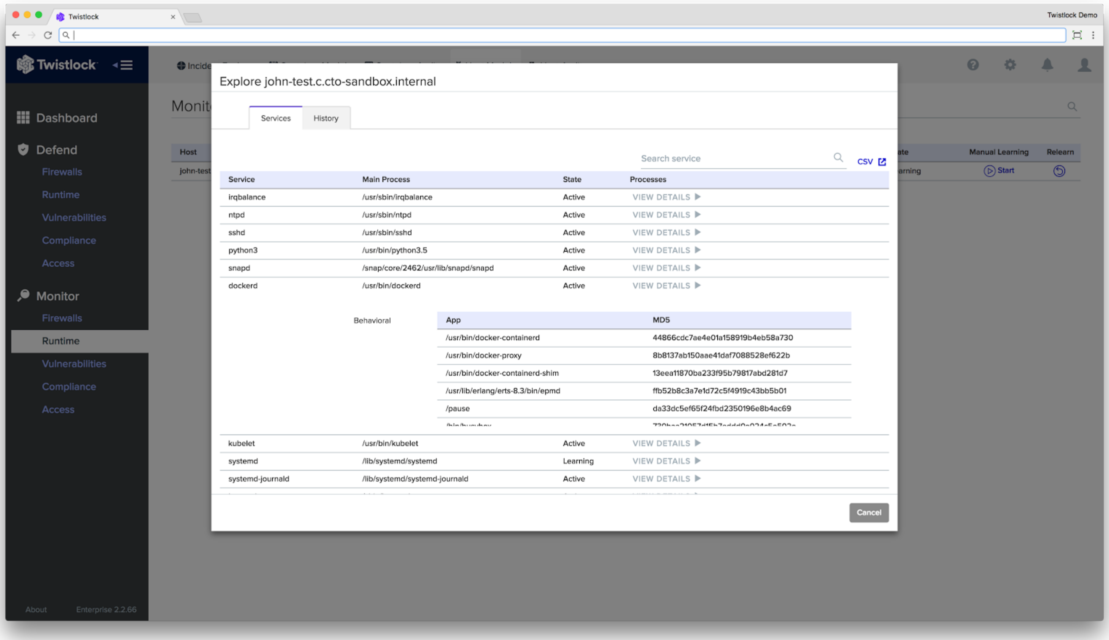
ID	Container	Image	Hostname	Date	Archive
#4	/k8s_rabbitmq_rabbitmq-3917772209-2hjqk_sock-s...	rabbitmq:3.6.8	john-test.c.cto-sandbox.internal	Sep 5, 2017 2:53:46 PM	
#1	/youthful_darwin	morello/httpd:latest	john-test.c.cto-sandbox.internal	Aug 30, 2017 1:56:50 PM	

Each of these actions would trigger a separate sensor in our runtime platform but Incident Explorer brings together the full set of actions into a clear flow, elevating them beyond simple data into actionable security intelligence. Rather than having to manually sift through data and timestamps, Incident Explorer builds this chain of events automatically, highlighting key indicators along the way and enabling more rapid and effective incident response.

Read more about Incident Explorer.

Runtime defense for container hosts

Our motto is cloud native cyber security, which means going beyond just containers to help secure the entire cloud native stack, from the apps they run to the hosts they run on. In 2.2, our promise is simple: Twistlock is the only security tool you need on a container host. We've long provided threat based runtime defense (IP reputation lists and malware signatures) to your hosts, but in 2.2, we're applying the same model driven architecture to protect hosts as well as containers.



Explore john-test.c.cto-sandbox.internal

Service	Main Process	State	Processes
irqbalance	/usr/sbin/irqbalance	Active	VIEW DETAILS ▶
ntpd	/usr/sbin/ntpd	Active	VIEW DETAILS ▶
sshd	/usr/sbin/sshd	Active	VIEW DETAILS ▶
python3	/usr/bin/python3.5	Active	VIEW DETAILS ▶
snapd	/snap/core/2462/usr/lib/snapd/snapd	Active	VIEW DETAILS ▶
dockerd	/usr/bin/dockerd	Active	VIEW DETAILS ▶
kubelet	/usr/bin/kubelet	Active	VIEW DETAILS ▶
systemd	/lib/systemd/systemd	Learning	VIEW DETAILS ▶
systemd-journald	/lib/systemd/systemd-journald	Active	VIEW DETAILS ▶

Behavioral

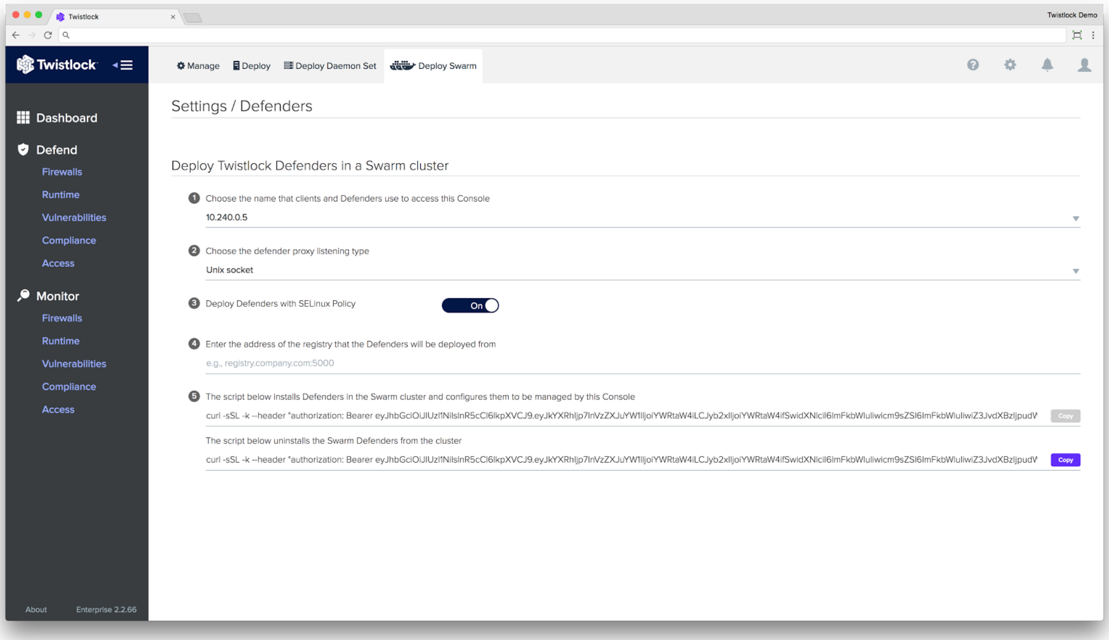
App	MD5
/usr/bin/docker-containerd	44866cdc7ae4e01a158919b4eb58a730
/usr/bin/docker-proxy	8b8137ab150aae41daf7088528ef622b
/usr/bin/docker-containerd-shim	13eea11870ba233f95b79817abd281d7
/usr/lib/erlang/erts-8.3/bin/epmd	ffb52b8c3a7e1d72c5f4919c43bb5b01
/pause	da33dc5ef65f24fbd2350196e8b4ac69

Obviously, a host is different than a container and is far more likely to change over time as it’s updated and upgraded. Thus, we’ve tuned the ML algorithms we use for building models to create models that are optimized for host protection. For hosts, we learn the system services that each hosts runs as non-interactive background processes. These models predict the specific binaries and checksums executed and the execution patterns of each and we disambiguate those processes run within a service from those run interactively to eliminate false positives. For example, if the dhcpd service on a host is compromised and leads to netcat spawning, we’ll detect (and optionally prevent) that. However, if an admin logs on through a properly authenticated ssh session and runs apt update `&& apt upgrade -y` all of the resulting processes spawned and file system changes observed are allowed because we correlate them with the genuine session in which they were created.

Read more about host runtime defense.

Native deployment on Swarm

In 2.0 and 2.1, we provided a native experience for deploying and running Twistlock as pods and Daemon Sets on Kubernetes. In 2.2, we’ve added similar capabilities for Docker Swarm. That means that if you’re running Swarm as your orchestrator, you can run Console as a service (and rely on Swarm for providing built in high availability) and Defender as a global service. So, whether your Swarm cluster has 5 nodes or 500 nodes, you can protect each of them with Twistlock with just a few clicks (or CLI commands) in just a couple of minutes.



Settings / Defenders

Deploy Twistlock Defenders in a Swarm cluster

1. Choose the name that clients and Defenders use to access this Console: 10.240.0.5
2. Choose the defender proxy listening type: Unix socket
3. Deploy Defenders with SELinux Policy: On
4. Enter the address of the registry that the Defenders will be deployed from: e.g., registry.company.com:5000
5. The script below installs Defenders in the Swarm cluster and configures them to be managed by this Console

```
curl -sSL -k --header "authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjp7InVzZXJuYW1lIjoiYWRtaW4iLCJyb2xlIjoiYWRtaW4ifSwidXNlcil6ImFkbWluIiwicm9sZSI6ImFkbWluIiwiZ3JvdXBzIjpudV
```

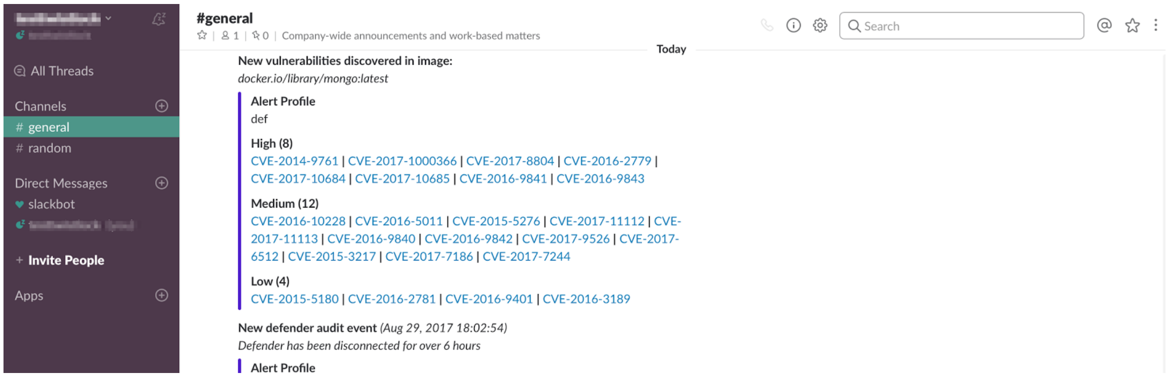
The script below uninstalls the Swarm Defenders from the cluster

```
curl -sSL -k --header "authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJkYXRhIjp7InVzZXJuYW1lIjoiYWRtaW4iLCJyb2xlIjoiYWRtaW4ifSwidXNlcil6ImFkbWluIiwicm9sZSI6ImFkbWluIiwiZ3JvdXBzIjpudV
```

Read more about native deployment on swarm.

Slack and JIRA push alerts

One of our guiding architectural principles has to make sure our data is open, accessible, and that we meet customers where they are rather than forcing them to come to us. Practically, that means we've always supported a diverse set of integrations for alerting and monitoring from getting JSON via our API, to detailed syslog output, to email alerting. In 2.2, we're making this even easier for customer using popular collaboration tools like Slack and JIRA. In previous releases, integration with these platforms was provided via email. While functional, it wasn't a native experience in those tools.



New vulnerabilities discovered in image:
docker.io/library/mongo:latest

Alert Profile
def

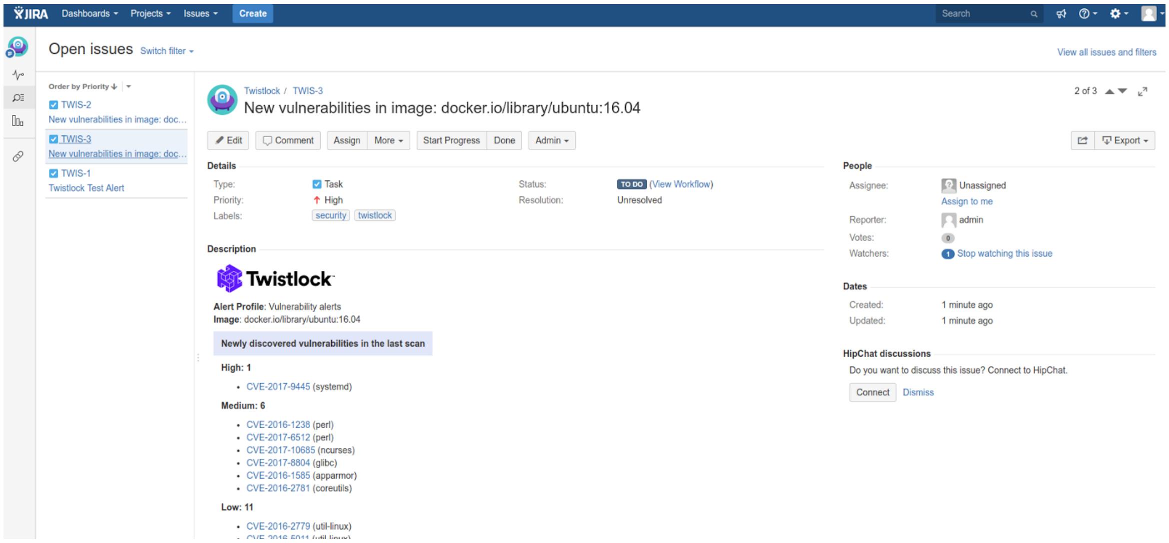
High (8)
CVE-2014-9761 | CVE-2017-1000366 | CVE-2017-8804 | CVE-2016-2779 | CVE-2017-10684 | CVE-2017-10685 | CVE-2016-9841 | CVE-2016-9843

Medium (12)
CVE-2016-10228 | CVE-2016-5011 | CVE-2015-5276 | CVE-2017-11112 | CVE-2017-11113 | CVE-2016-9840 | CVE-2016-9842 | CVE-2017-9526 | CVE-2017-6512 | CVE-2015-3217 | CVE-2017-7186 | CVE-2017-7244

Low (4)
CVE-2015-5180 | CVE-2016-2781 | CVE-2016-9401 | CVE-2016-3189

New defender audit event *(Aug 29, 2017 18:02:54)*
Defender has been disconnected for over 6 hours

Alert Profile



- ☑ TWIS-2 New vulnerabilities in image: doc…
- ☑ TWIS-3 New vulnerabilities in image: doc…
- ☑ TWIS-1 Twistlock Test Alert

Twistlock / TWIS-3

New vulnerabilities in image: docker.io/library/ubuntu:16.04

Details

Type:	☑ Task	Status:	TO DO (View Workflow)
Priority:	↑ High	Resolution:	Unresolved
Labels:	security twistlock		

People

Assignee:	Unassigned Assign to me
Reporter:	admin
Votes:	0
Watchers:	1 Stop watching this issue

Dates

Created:	1 minute ago
Updated:	1 minute ago

HipChat discussions

Do you want to discuss this issue? Connect to HipChat.

Description

Twistlock

Alert Profile: Vulnerability alerts
Image: docker.io/library/ubuntu:16.04

Newly discovered vulnerabilities in the last scan

High: 1

- CVE-2017-9445 (systemd)

Medium: 6

- CVE-2016-1238 (perl)
- CVE-2017-6512 (perl)
- CVE-2017-10685 (ncurses)
- CVE-2017-8804 (glibc)
- CVE-2016-1585 (apparmor)
- CVE-2016-2781 (coreutils)

Low: 11

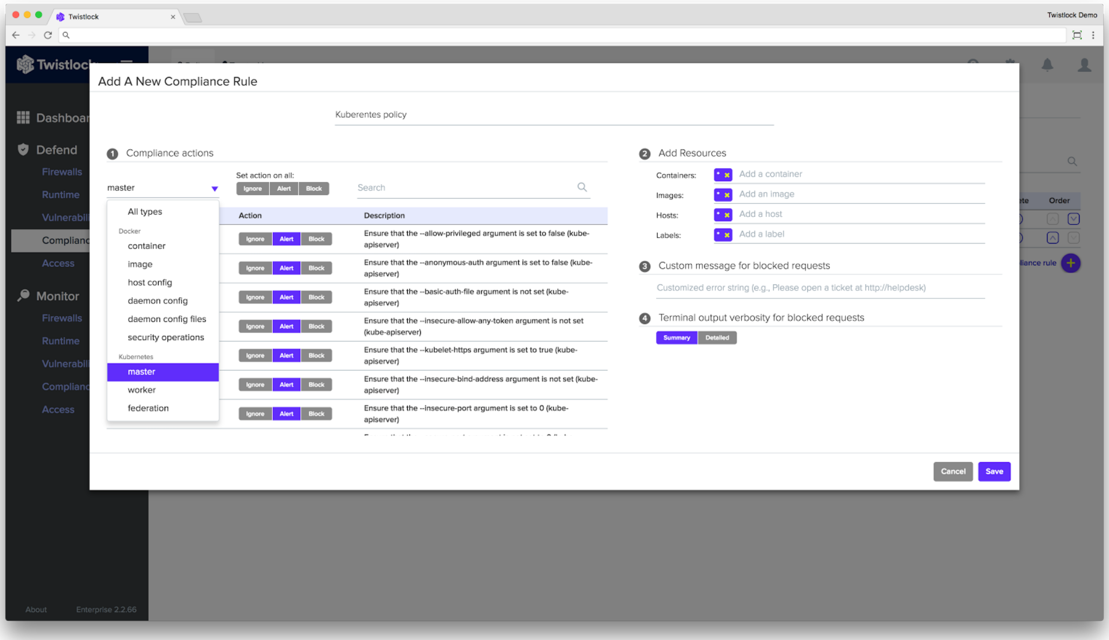
- CVE-2016-2779 (util-linux)
- CVE-2016-5011 (util-linux)

In 2.2, we have Slack and JIRA integration in the box and setting it up is as simple as providing a few details about the channel or JIRA server you want to push alerts to. Additionally, we've centralized and simplified alert profiles into a single place in the Console. You still have all the control and granularity to define different providers and targets per rule but you can now assign all these from a central alert configuration page. If you want to get started fast with a simple push alert config, you can now do that too, by just pointing all alerts for a given product area (like Vulnerabilities or Runtime) to whatever Slack channel, JIRA project, or email addresses you want to send them to.

Read more about new push alert providers.

Compliance monitoring and enforcement for Kubernetes

We're proud to have contributed to the Kubernetes CIS Benchmark, which builds on our many other compliance focused contributions in the community, like NIST SP 800-190 and our guides for PCI and HIPAA in containerized environments. While many other individuals and companies contribute to these works, our approach in how we 'productize' compliance is significantly different. Rather than simply provide a standalone scanning tool or script, or provide a small number of built in checks in the product, we look at compliance as a critical part of the platform. Not only is it a regulatory necessity for many of our customers, it also provides critical layers of security defense in depth. Thus, for us, compliance is a comprehensive set of configurable rules to assess, monitor, and enforce adherence to the controls important to your organization.



Add A New Compliance Rule

Kuberentes policy

1 Compliance actions

master

Set action on all: Ignore Alert Block

Action	Description
Alert	Ensure that the --allow-privileged argument is set to false (kube-apiserver)
Alert	Ensure that the --anonymous-auth argument is set to false (kube-apiserver)
Alert	Ensure that the --basic-auth-file argument is not set (kube-apiserver)
Alert	Ensure that the --insecure-allow-any-token argument is not set (kube-apiserver)
Alert	Ensure that the --kubelet-https argument is set to true (kube-apiserver)
Alert	Ensure that the --insecure-bind-address argument is not set (kube-apiserver)
Alert	Ensure that the --insecure-port argument is set to 0 (kube-apiserver)

2 Add Resources

Containers: * Add a container

Images: * Add an image

Hosts: * Add a host

Labels: * Add a label

3 Custom message for blocked requests

Customized error string (e.g., Please open a ticket at http://helpdesk)

4 Terminal output verbosity for blocked requests

Summary

In 2.2, we've added support for all 106 settings in the Kubernetes CIS benchmark. This means that you can not only assess your organization's compliance with these recommendations, but also actively enforce them such that Twistlock will block configurations and deployments not allowed by your policy. Even more valuable, though, is that our Twistlock Labs research team analyzed each of the sections in the Kubernetes CIS Benchmark and scored them based on criticality. While all of the settings have merit, we analyzed the ones of most criticality to operating your cluster and apps securely and provide clear guidance to help you prioritize these settings.

Of course, all these new compliance capabilities for Kubernetes are integrated into Compliance Explorer, so you have a simple place to monitor compliance across all your environments.

Read more about Kubernetes compliance features.