Securing Container Image Supply Chains with tools such as Goss and OpenSCAP

Chris Howarth - Citi

AGENDA

- Background – Securing Container Images
- Introduce Goss and OpenSCAP for Security Validation Tools (+ demo)
- Image Vulnerability Scanning
- Securing container supply chain using end to end pipeline
- Key successes

BACKGROUND – SECURING CONTAINER IMAGES

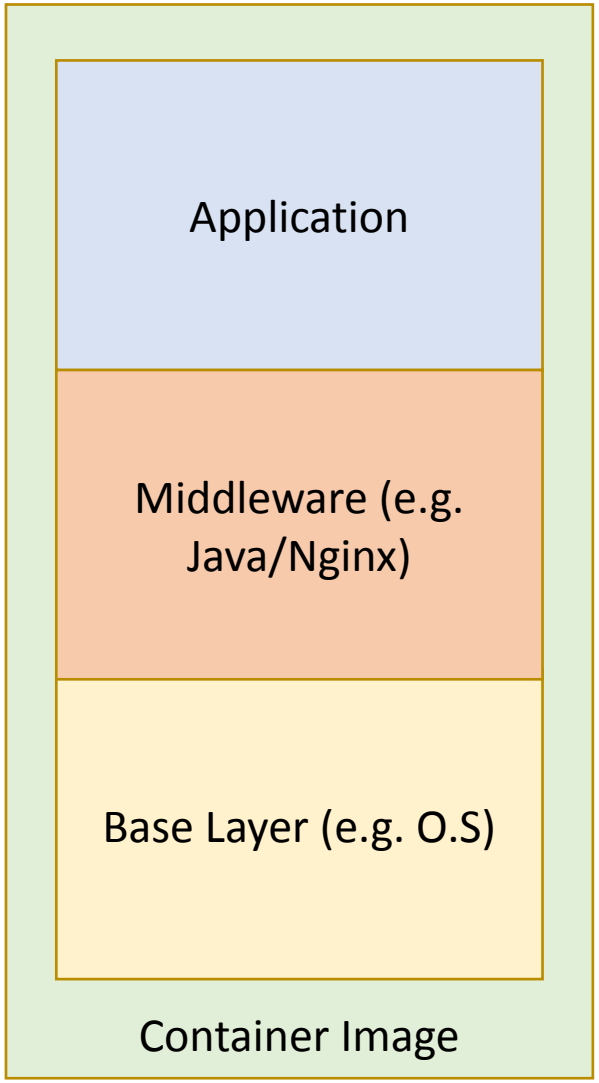
- Container Image Adoption Continues to Grow Steeply:

  Gartner – Jun 2020 – "*Gartner predicts that by 2022, more than 75% of global organizations will be running containerized applications in production, up from less than 30% today.*"

- Most (if not all) organizations consume external 3rd party images
  - How do we know to trust these external base images ?
  - Need trust final image after additional layers added

- Software Supply Chain Attacks
  - Codecov (Apr 2020) – bash uploader script modified by infiltration of Docker image creation process
  - Sunburst/Solarwinds (2019/20) – malicious code injection planted by hackers into Orion IT monitoring system code

- How to minimize the risk of consuming 3rd party images ?



CONTAINER IMAGE SECURITY VALIDATION TOOLS

Two Open Source Tools

- **Goss** https://github.com/aelsabbahy/goss
- **Openscap** https://www.open-scap.org/

What sort of checks do these tools provide ?

- File and package integrity checks
- World writable files/dirs.
- Suid checks
- Presence of packages (wanted or unwanted)
- Full compliance checks per benchmarks such as CIS/ DoD STIG
- Customized checks

DEMO

SAMPLE INSECURITIES DETECTED

- World writable files and directories
- Shell access for application accounts/functional IDs
- Careless PATH settings (see below)

CIS Benchmark:
6.2.6 Ensure root PATH Integrity (Scored)
Profile Applicability:
- Level 1 - Server
- Level 1 - Workstation

Description:
The root user can execute any command on the system and could be fooled into executing programs unintentionally if the PATH is not set correctly.

Rationale:
Including the current working directory (.) or other writable directory in root 's executable path makes it likely that an attacker can gain superuser access by forcing an administrator operating as root to execute a Trojan horse program.

```
# echo $PATH
/opt/myapp/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/s
bin:/bin

# ls -ld /opt/myapp/bin
drwxr-xr-x. 2 testuser root 18 Jul 16 12:01 /opt/myapp/bin

# cat /opt/myapp/bin/rpm
#!/bin/bash
echo "hello there"

# rpm
hello there

# which rpm
/opt/myapp/bin/rpm
```

COMBINED BUILD AND TEST OF IMAGE – PUTTING IT ALL TOGETHER

- **Use Multi-stage Docker Build**

```
$ cat Dockerfile
##########################################
# Parent image
##########################################
FROM <path_to_raw_o.s._image> as hardened

# Add in customizations
COPY mycert /certificate /certdir
……..

# Harden image
RUN userdel games
…..

# Save final image
FROM hardened as validation

# Run Goss tests
RUN dnf install goss –y
COPY goss.yaml /
RUN goss validate

# Now run OpenSCAP steps
RUN dnf install openscap-scanner scap-security-guide –y
RUN oscap xccdf eval --results results.xml --report results.html
--profile cis /usr/share/xml/scap/ssg/content/ssg-rhel8-xccdf.xml


# Save the hardened image
FROM hardened as final
```

CONTAINER IMAGE SECURITY VALIDATION TOOLS

Goss

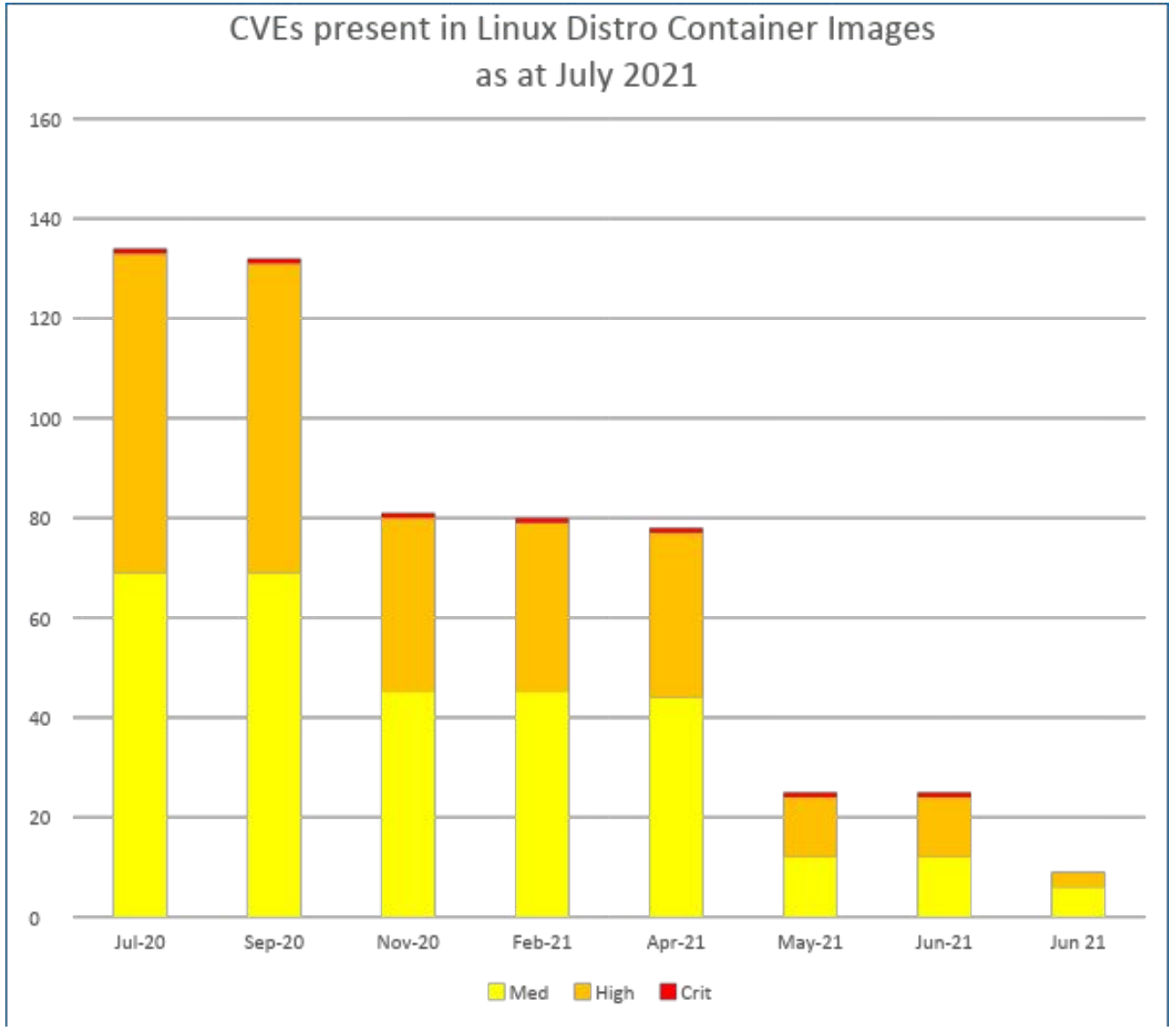
- Very simple to install (single statically Golang binary)
- Linux flavour independent
- Extremely simple to take a secure container as a template and generate policy rules
- Easy to create new custom checks

OpenSCAP

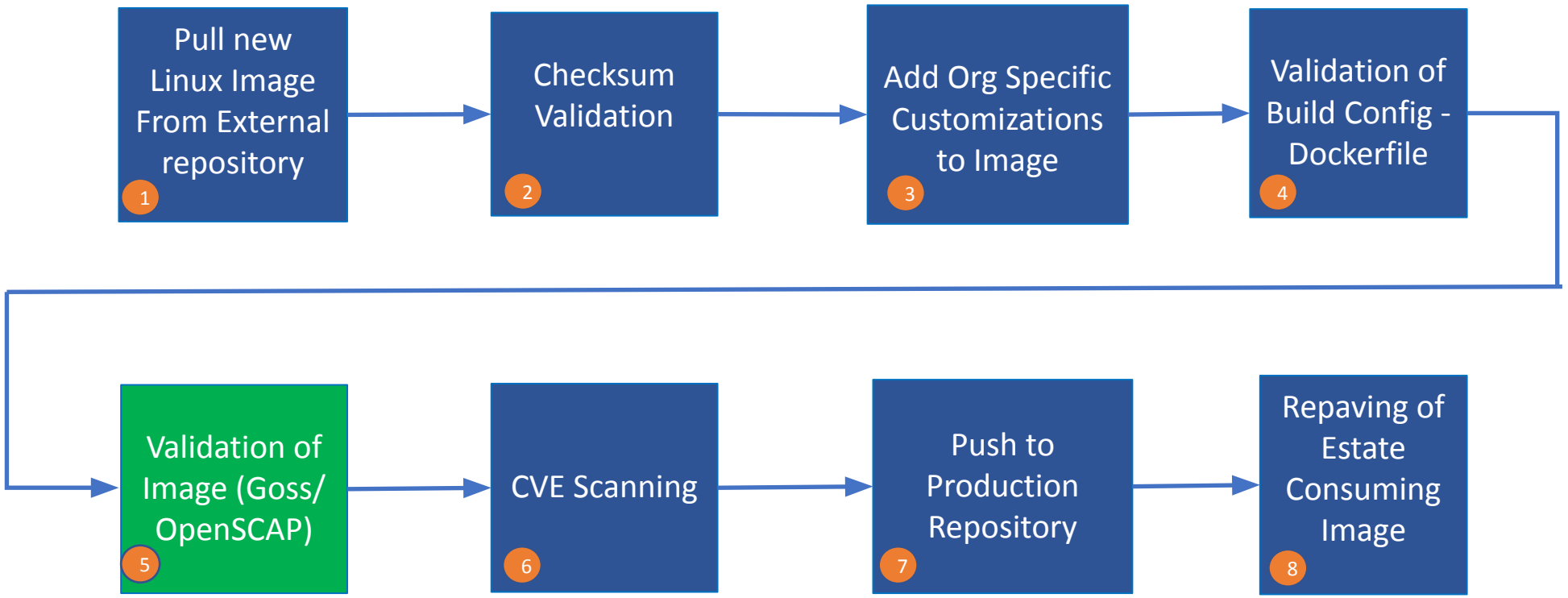
- Simple to install and run
- Contains pre-canned sets of policies e.g. CIS compliance checks, DoD STIG
- Produces the human readable HTML reports
- Includes a workbench tool to create custom policies

CVE SCANNING OF IMAGES

- Always use the latest image available!
- Rebuild and repave images as frequently as possible to minimize exposure to CVEs

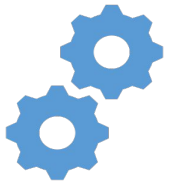


EXAMPLE BASE IMAGE INGESTION PIPELINE



- **Key Consideration:**
  - Use podman rather than docker to build images

KEY SUCCESSES



Full automation

No human involvement – ensures reproducibility, more secure and fast

- Images released to production repos same day as availability from 3rd party source
- Adoption of the very latest images significantly minimizes CVE vulnerabilities

All pipeline components themselves have automated testing – ensures pipeline is robust and speeds development

Secured Containers

Higher degree of trust in images

- Tooling such as Goss/Openscap validate every image
- Imported images automatically scan for CVEs

Questions