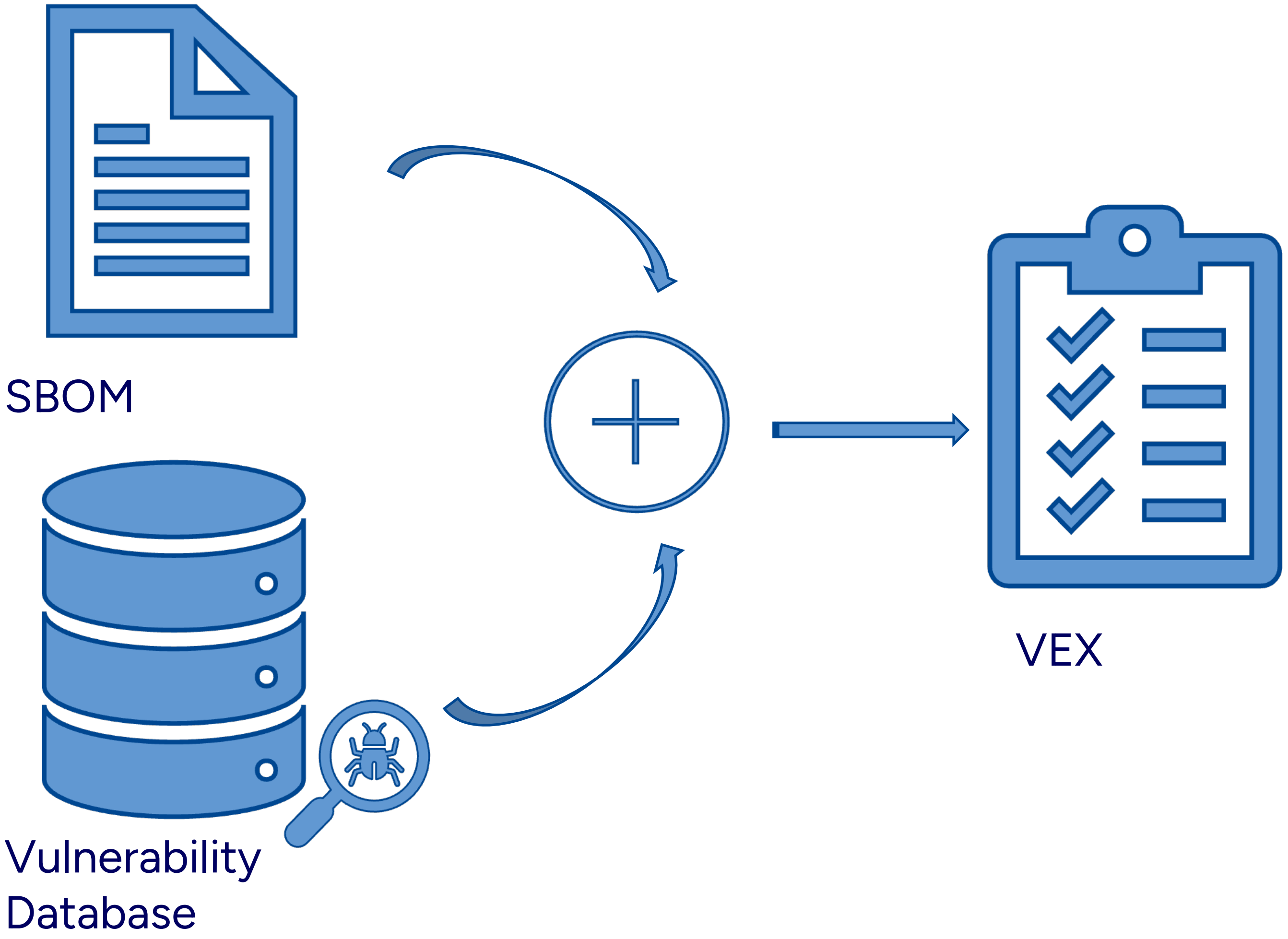
VEX-generation for containers

Presented by: Yekatierina Churakova
PhD Student in KTH
CHAINS project researcher

VEX (Vulnerability Exploitability eXchange): overview



VEX: key components

Vulnerability Database

Exploit Database

Exchange Mechanizm

Ref: Minimum Requirements for Vulnerability Exploitability eXchange (VEX) (cisa.gov)

VEX: Tools list

aqua
trivy

grype

VEXY

OSV

docker

snyk

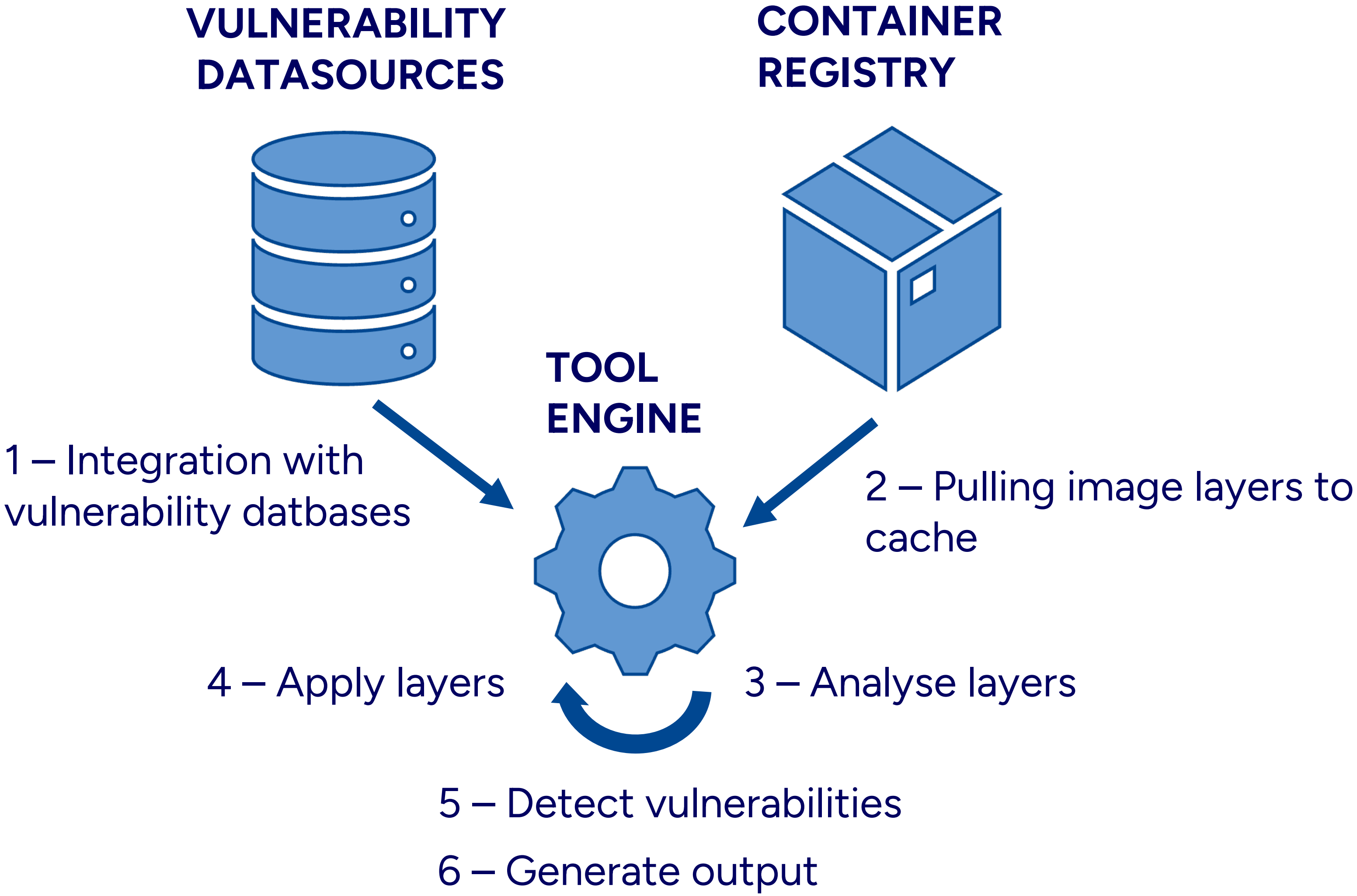
falco

DEPSCAN

OpenSCAP

clair

VEX: production



VULNERABILITY DATASOURCES

CONTAINER REGISTRY

TOOL ENGINE

1 – Integration with vulnerability datbases

2 – Pulling image layers to cache

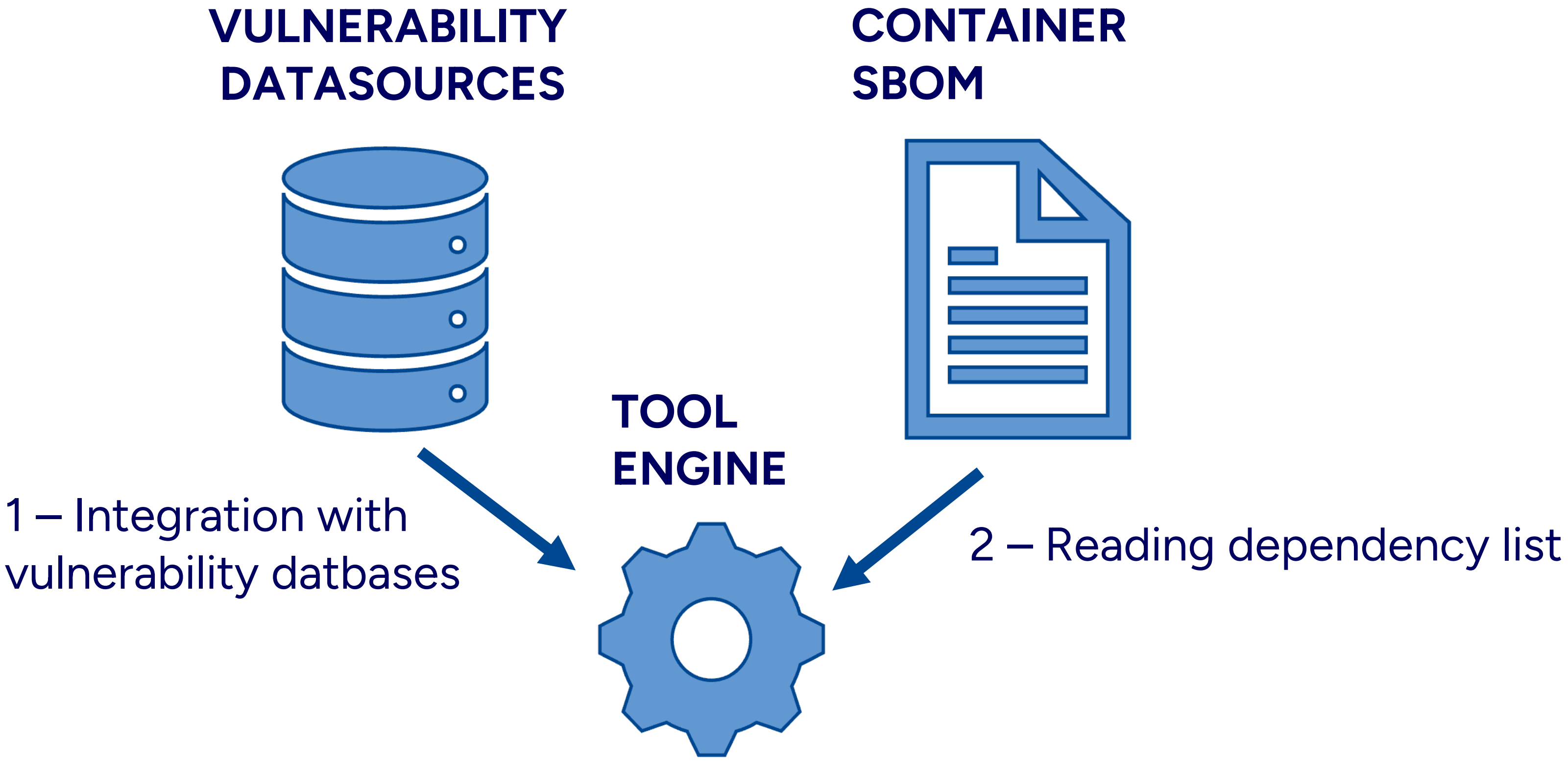
3 – Analyse layers

4 – Apply layers

5 – Detect vulnerabilities

6 – Generate output

VEX: production, alternative way



VULNERABILITY DATASOURCES

CONTAINER SBOM

TOOL ENGINE

1 – Integration with vulnerability datbases

2 – Reading dependency list

3 – Map vulnerabilities to dependencies

4 – Generate output

VEX: results

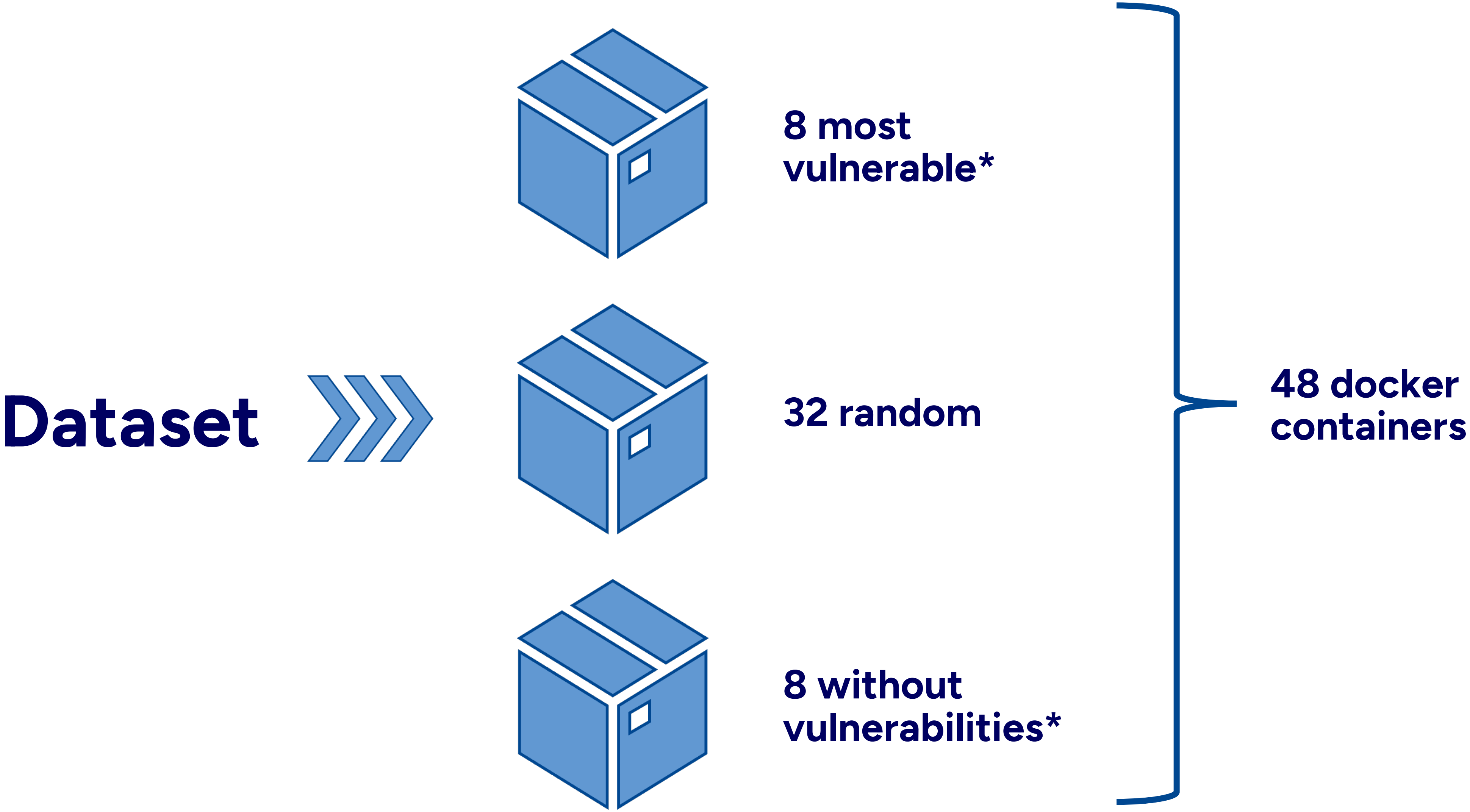
	Trivy	Grype	DepScan	OSV	Vexy	Docker scout	Clair	Dagda	Snyk	OpenScap	Falco
Scans SBOMs	+	+	+	+	+	-	-	-	(+)-	-	-
Scans docker image	+	+	+	-	-	+	+	+	+	+	+
Produces SBOMs	+	+	+	-	-	+	-	-	(+)-	-	-

Vulnerability grading scales

- Docker: Critical, High, Medium, Low, **Unspecified**
- Grype: Critical, High, Medium, Low, **Negligible**
- Trivy: Critical, High, Medium, Low
- Vexy: Critical, High, Medium, Low
- OSV: Critical, High, Medium, Low, **Unrated**
- DepScan: Critical, High, Medium, Low
- Snyk: Critical, High, Medium, Low
- Clair: Critical, High, Medium, Low
- Falco: Critical, High, Medium, Low
- OpenScap: Critical, High, Medium, Low
- Dagda: Critical, High, Medium, Low

Hypothesis

Wouldn’t it be reasonable to think that all tools produce the same output for a same container?



Dataset

8 most vulnerable*

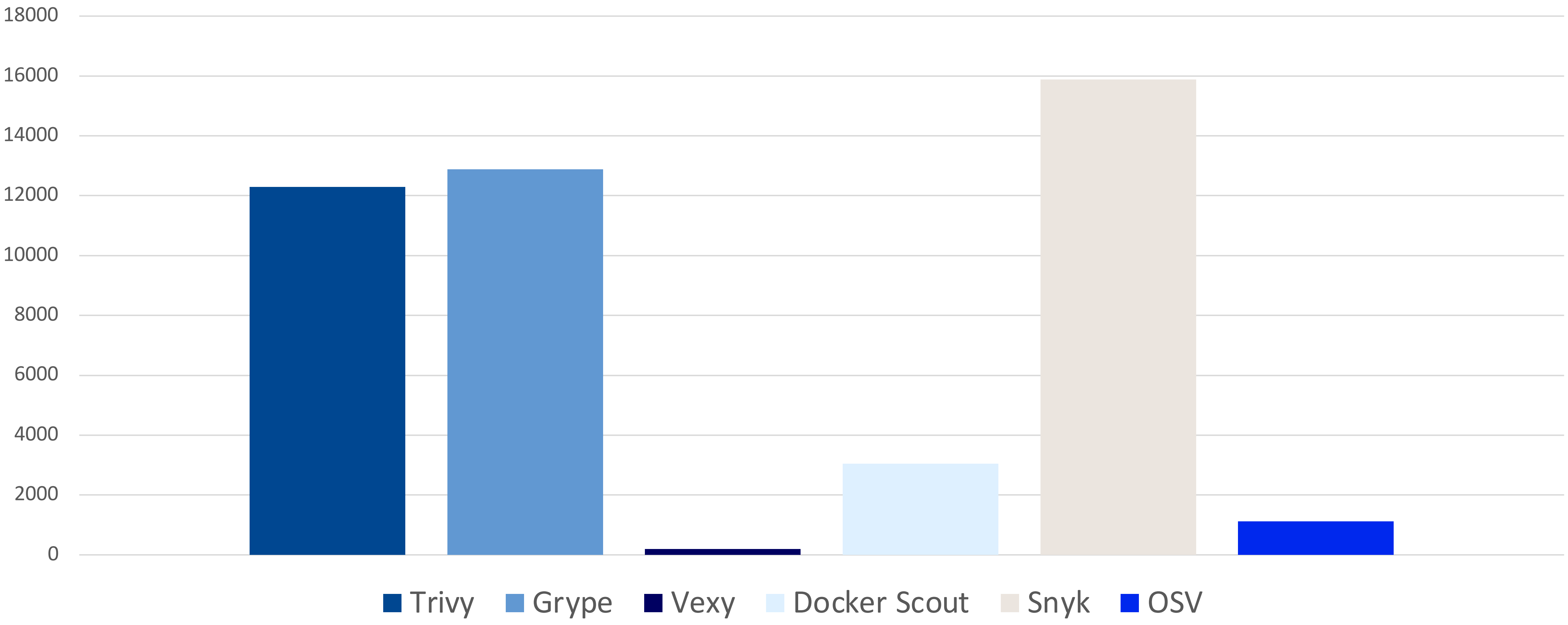
32 random

8 without vulnerabilities*

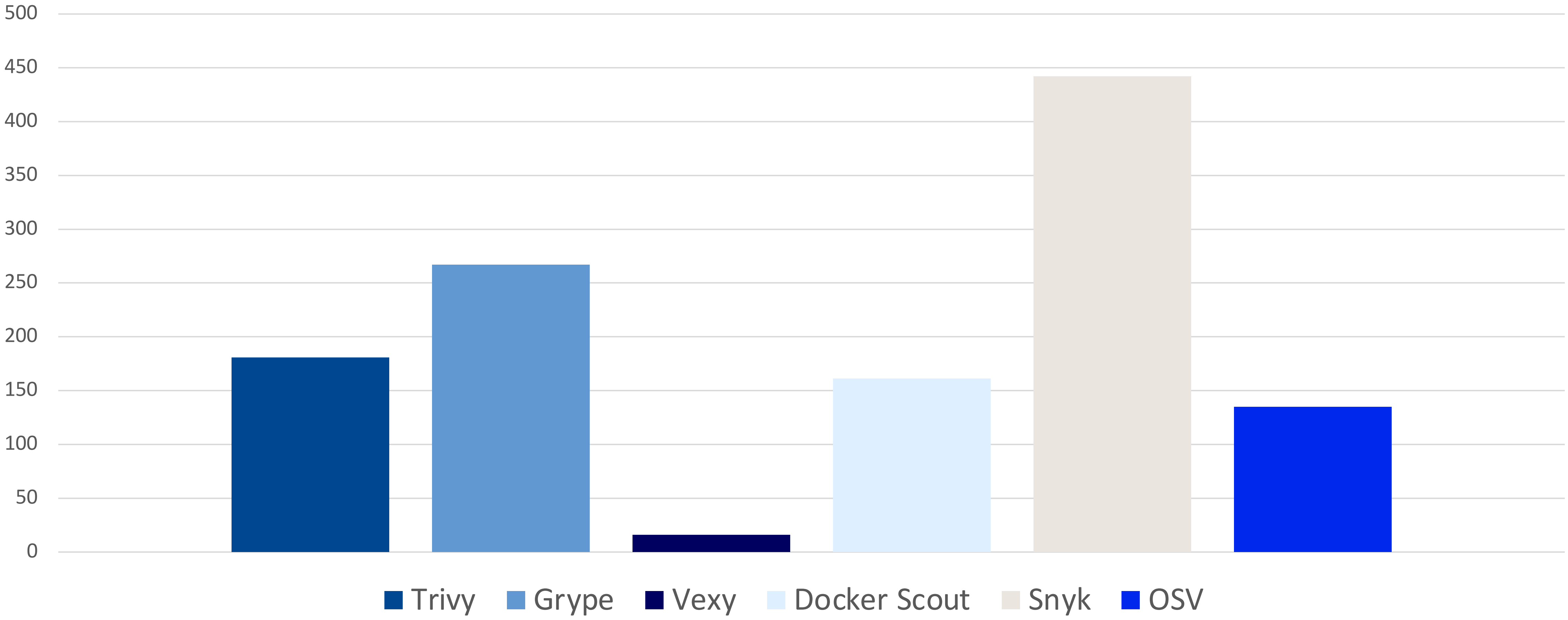
48 docker containers

*according to docker hub

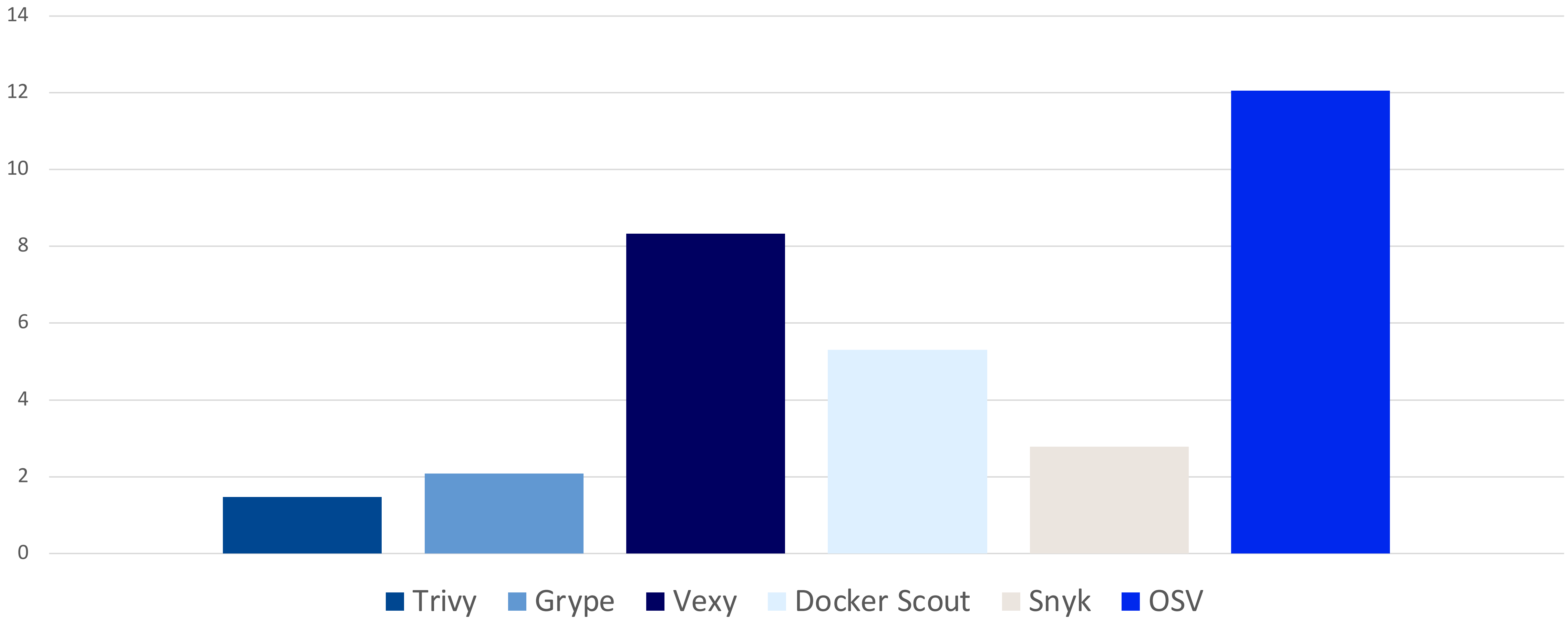
Number of total vulnerabilities per tool



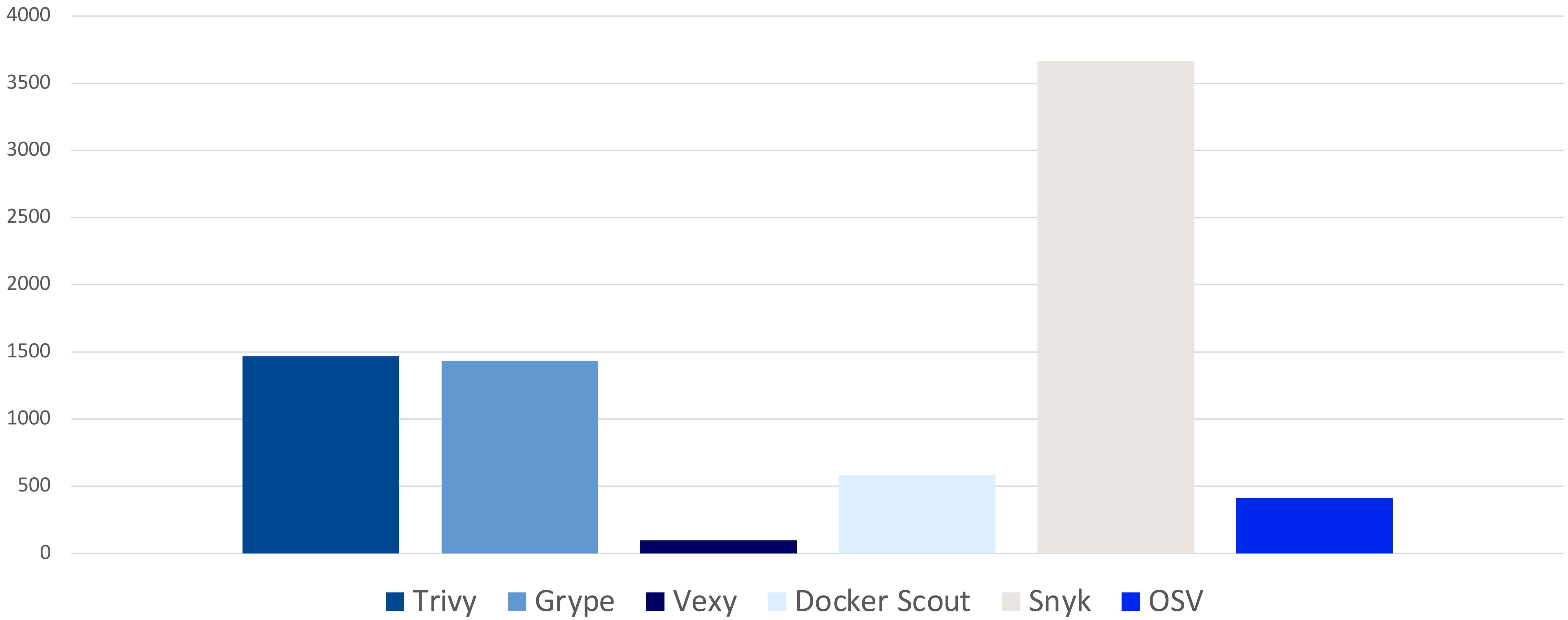
Number of Critical vulnerabilties



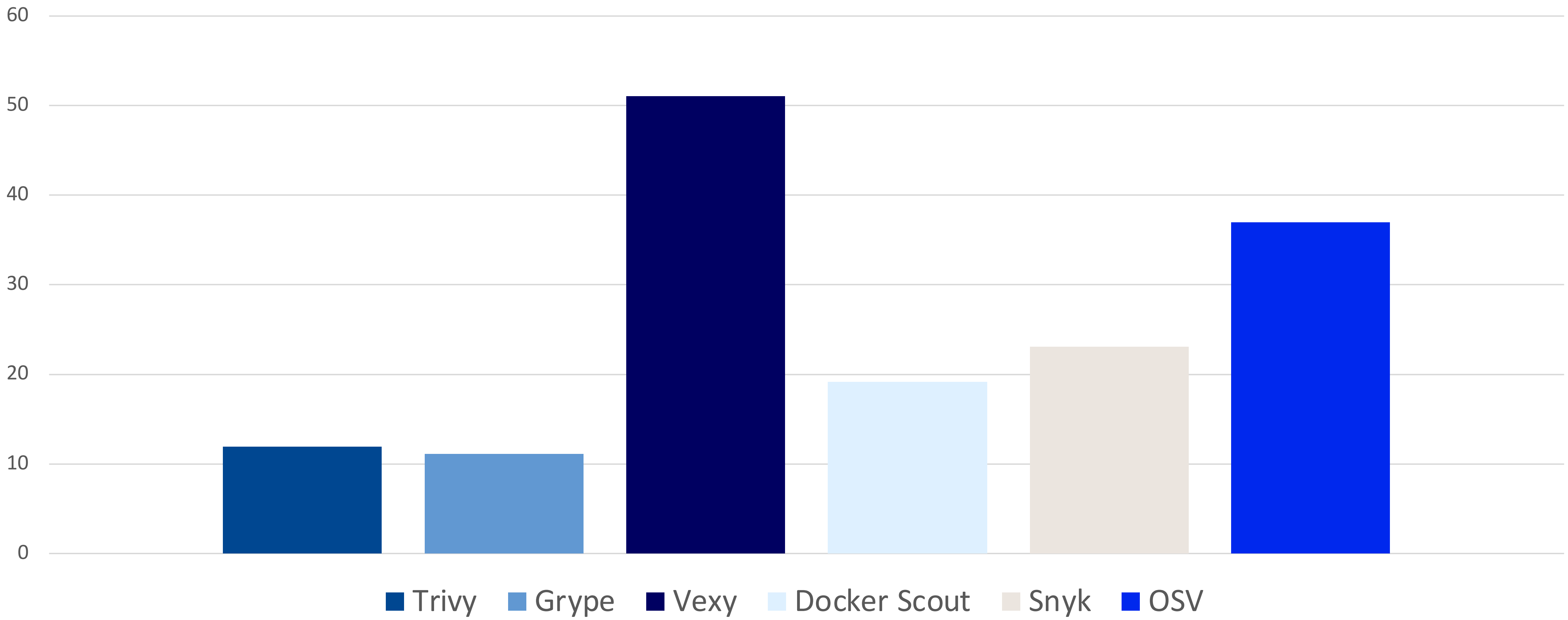
Percentage of Critical vulnerabilities



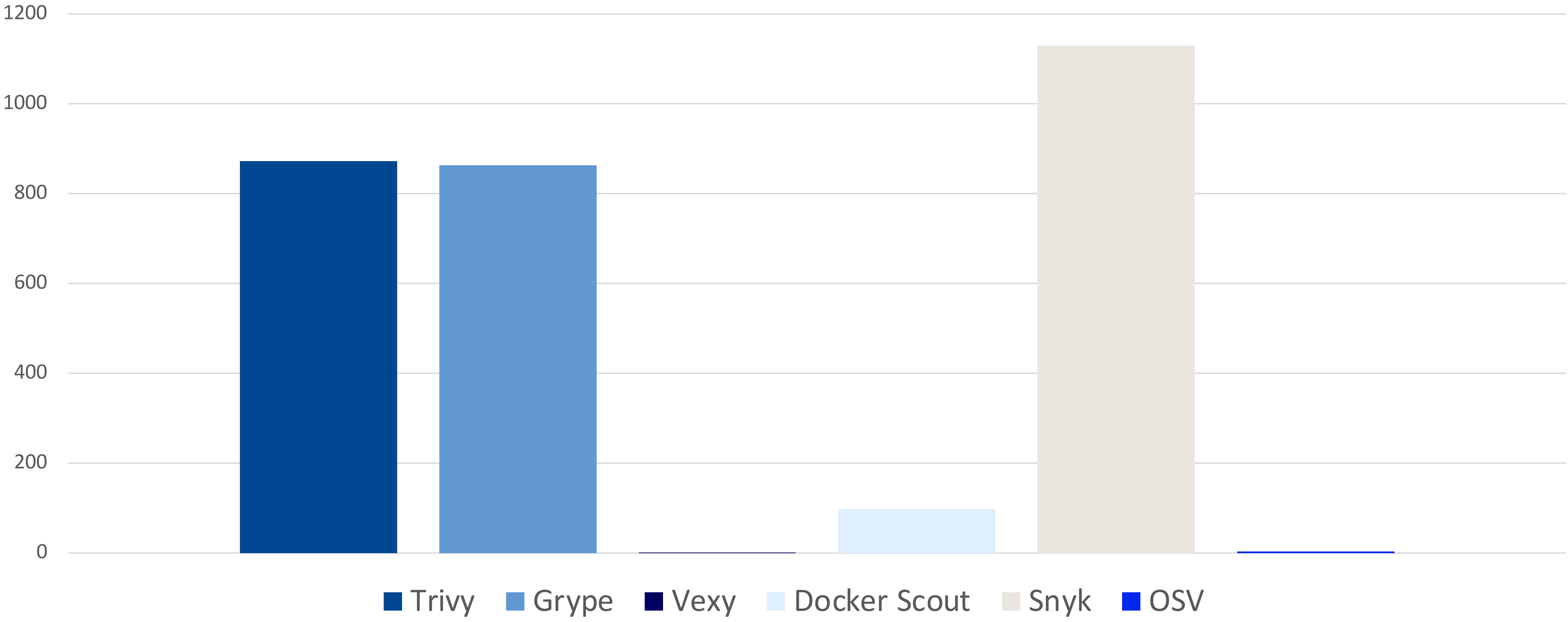
Number of High vulnerabilties



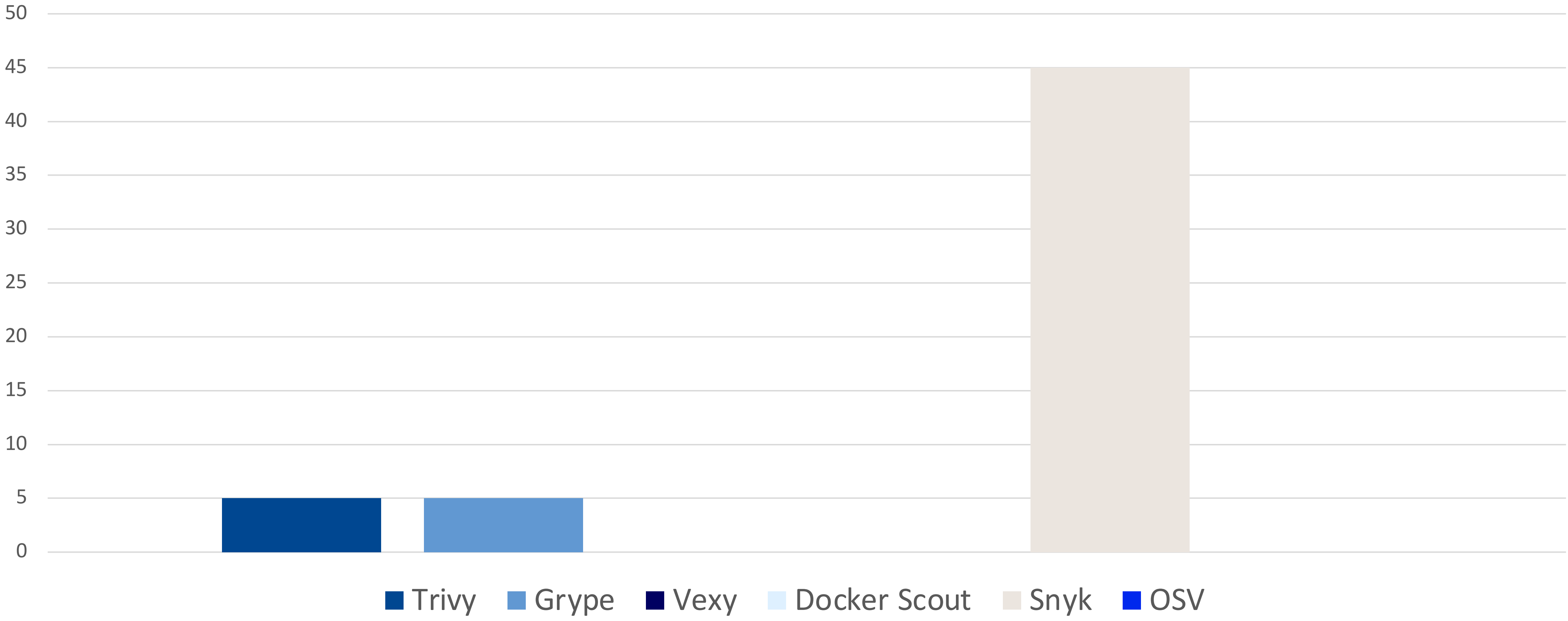
Percentage of High vulnerabilities



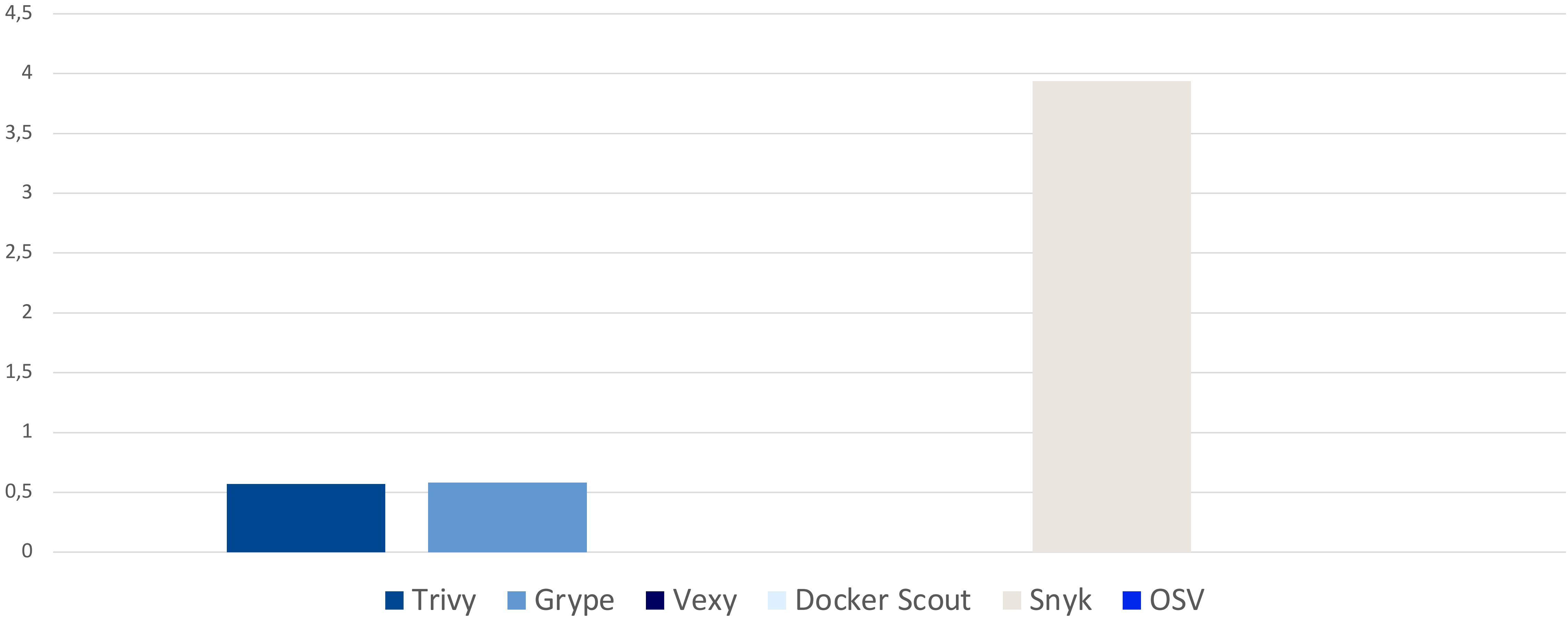
Number of total vulnerabilties per tool (in example of single container ruby:latest)



Number of Critical vulnerabilities (in example of single container ruby:latest)



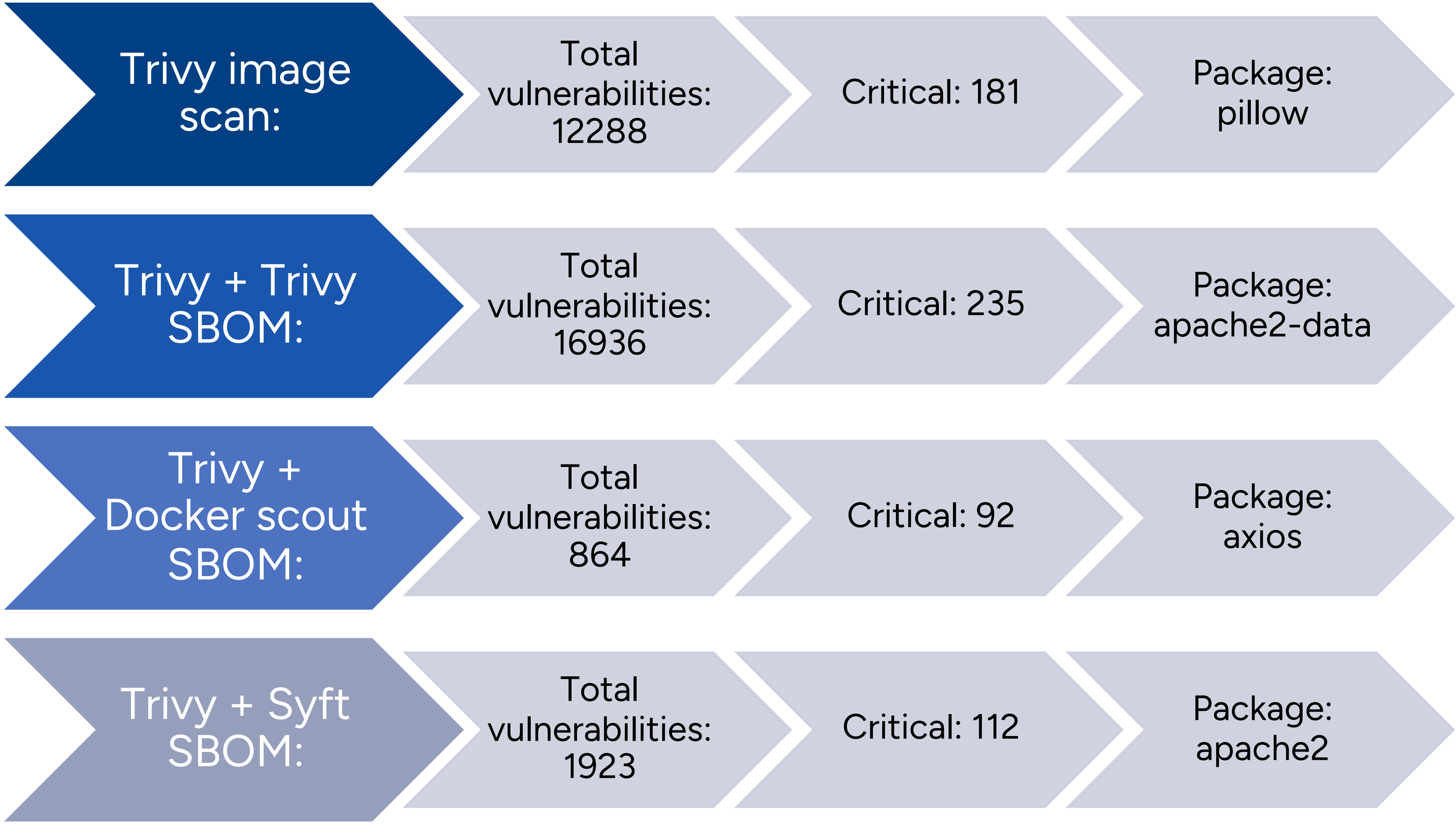
Percentage of Critical vulnerabilities(in example of single container ruby:latest)



Most vulnerable packages

Grype	• bsdutils
Trivy	• pillow
OSV	• axios
Docker Scout	• openssl@3.14
Vexy	• -------
Snyk	• apache22.4.57-2

Output difference (example of Trivy)



Trivy image scan:	Total vulnerabilities: 12288	Critical: 181	Package: pillow
Trivy + Trivy SBOM:	Total vulnerabilities: 16936	Critical: 235	Package: apache2-data
Trivy + Docker scout SBOM:	Total vulnerabilities: 864	Critical: 92	Package: axios
Trivy + Syft SBOM:	Total vulnerabilities: 1923	Critical: 112	Package: apache2

Further work

- **Test tools on scaning other container formats:**
  - OCI-compliant images
  - Tar-archives
  - Singularity images

- **Test tools on SBOMs for various container formats:**
  - OCI-compliant images
  - Tar-archives
  - Singularity images

- **Deeper analysis:**
  - Other metrics with variance
  - Measurements

VEX: preliminary conclusion

1. VEX is a good way to monitor the security of a new build or release.

2. CHAINS project like the concept of VEX:)

3. VEX-producing tool should be carefully chosen.

4. Inititial recommendation: to focus on tools, which have regular updates.