0 intro

- today im gonna tell u about automating security aspects of delivery pipeline
- wojtek – thanks for invitation!
- Guys – please do ask questions during my talk!

1 intro

- About me

3 Agenda

- not a rocket science but rather an invitation for playing with automated security testing
- We will talk about Jenkins, ansible, owasp zap and other security tools

3 Delivery pipeline

- how many devops, pentesters, security engineers?
- Nice distribution out there!
- Goal of Delivery pipeline is to make sure that commited changes are of good quality
- Linux containers: tiny, little isolated linux environments we can start and stop immediately

Manual testing demo #1.1

- OWASP WebGoat: java app, so won't use this as we dont want to celebrate 20 years of out of memory error
- Hackademic like WebGoat is testing ground
  - Unfortunately not working – had to fix it
  - Mention google summer of code
  - There is a playbook for hackademic in repo

Manual testing demo #1.2

- So instead I created couple of apps
- But then decided that we don't want to learn about vulnerabilities, but rather about automation
- So we've got “noapp” there (see urself sqli.php)
- There is nothing good with this app – oh it works
- Show webpage
- Run OWASP ZAP – from scratch with param
- Run simple test on sqli
- Fix sqli problem and deploy_vulnerable
- header('X-Frame-Options: DENY');

Manual testing demo #2

- Downloaded jenkins and dependency check
- EEE_DEMO_2
- ./dependency-check.sh --noupdate --project test_jenkins --format XML --out . --scan ../jenkins/usr/lib/jenkins
- WAR – WebApplicationARchive
- NVD National Vulnerability Database
- National Institute of Standard and Technology

Automation tools

- Ansible: simple, easy to learn, crush complexity
- Show ansible galaxy
- Jenkins: scheduler with UI and many plugins for CI
- GoCD – like Jenkins, but instead of jobs it focuses on whole pipelines and provides many other high-level functionalities (maven vs ant)
- Actually GoCD performed not that well on my lab setup so sorry – not this time

Automated testing demo #1

- Show jenkins playbooks
- Run this playbook
- Show running Jenkins
- http://vm-jenkins.owasp.eee:8080/

Docker?

- About Linux containers (solaris zones, lxc, lxd, systemd-nspawn)
- With containers we can create separated Linux installations within seconds
  - With own libraries
  - With own users
  - e.g. only for one app
  - Docker provides only 1 process per container!

Docker registry?

- Simply a containers' images repository
- With git semantic (push/pull)
- We can create our own private repo!

Automated testing demo #2

- Deploy docker host:

- ansible-playbook -i inventory/all plays/deploy_docker.yml --tag docker-engine
- Systemctl status docker
- Docker search owasp
- Docker pull …
- Docker images
- Run scan from vm-docker by creating new container: alias EEE_DEMO_2
- Docker rm zap I pokazać na stable
- Zap-cli Dockerfile + run EEE_DEMO_3
- pip install python-owasp-zap-v2.4

Automated testing demo #2

- Show zap python api project
- Opowiedzie o problemie z apikey I python api

Automated testing demo #3

- Pokazać deploy docker-registry
- deploy_docker –tag docker-registry
- Wyjaśnić czemu statefull raczej w Vmce a nie w kontenerze dockerowym (ew. Systemd-nspawn lub lxc)
- Mention registryv1 and vs (distribution)
- Docker images
- Docker tag <ID> 127.0.0.1:5000/zap-cli
- Docker push 127.0.0.1:5000/zap-cli
- Searching? Problems with api:
  - docker search 127.0.0.1:5000/zap
  - curl -X GET http://localhost:5000/v1/search?q=zap
  - ls /var/lib/docker-registry/repositories/library/

Automated testing demo #3

- Pokazać I uruchomić EEE_DEMO_4
- Zainstalować dependency check:
- wget http://192.168.122.14/dependency-
- check-1.3.1-release.zip
- /opt/dependency-check/test, unzip,
- mkdir /opt/dependency-check/scan-jobs
- Exit, docker ps -a,
- docker commit <id> owasp/dependency-check-new
- docker tag <im_id> 127.0.0.1:5000/dependency-check
- docker push 127.0.0.1:5000/dependency-check-new
- EEE_DEMO_5

Automated testing demo #4

- Pokazać czystą instalację jenkinsa na Vmce
- Pokazać playbooka I opisać
- Uruchomić deploy_jenkins z tagiem
- Wskoczyć w jenkinsa I puścić oba testy I pokazać wyniki
- Omówić
  - dependency check trwa długo – może daily job?
  - Skanowanie zapem ma sens gdy targeted url

Docker inside docker pros and cons

- Omówić architekturę jenkinsa
- Jenkins nodes as docker containers?
- Should jenkins nodes run own docker containers
- This might be ok, but it's complicated
- Maybe instead some docker orchestration? PaaS?

What about false negatives / positives?

- We never know ;)
- We might use other tools to confirm (ask another doctor about your illness)
- We might re-run tests
- We might code our way out via creating database and scoring system
- SLIDE
- And this is what OWASP Benchamrk is all about
- SLIDE
- It's not rdy yet ;)

Summary

- Scanning on every change? Maybe with other tools or in narrower spectrum
- Security-scanning rather on daily / hourly basis
- Good compromise? deploy unsecure and fix it next day - that's for CD
- It's easy with targeted URLs.. but when we're spidering... it's time-taking
- SLIDE
- Classical pentesting is just must-have (or bug-bounty). Do you think that automation tools would find 0-days?