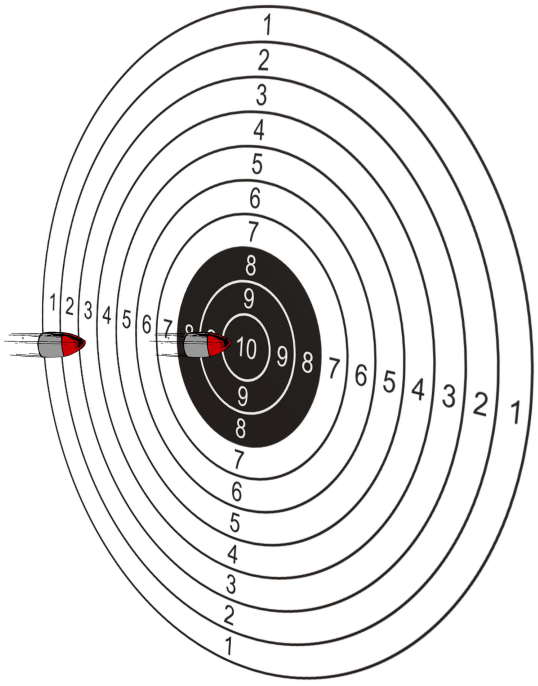
Open-Sec : Security Breakers

Walter Cuestas
OffSec Research Lead
atorres@open-sec.com

Quiénes somos ?

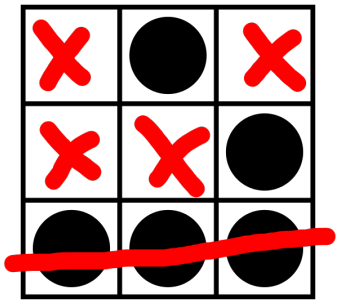
- Open-Sec
  - Más de 16 años
- Open Sec LLC
  - Desde el 2017 con clientes en Argentina, Chile, Bolivia, Ecuador, Colombia/El Salvador, España/Pakistán
  - Enero 2023 inicia operaciones en Panamá



Solamente hacemos Seguridad Ofensiva


HACKERS
CHALLENGE
blackhat

Qué hacemos ?



- **PENTESTING** :
  - A nivel de Centro de Datos tradicional (on-premises).
  - A nivel de Infraestructura, Software y Plataforma como Servicio (cloud).
  - A nivel de Aplicaciones (DAST, SAST) [Frontend y Backend].
  - A nivel de servicios de pago (comercio, pasarela, procesador, emisor, adquiriente).
  - A nivel de Sistemas de Control Industrial (ICS).
- **Pruebas de Seguridad para**
  - Cajeros Automáticos (ATMs) y Puntos De Venta Reales/Virtuales (Pos) .
  - Switches y Autorizadores Transaccionales.
- **Pruebas de Ingeniería Social e Intrusión Física.**
- **Revisión de la Seguridad del Código Fuente de Aplicaciones (cliente/servidor, web, móviles, web services, micro servicios).**
- **RED TEAMING.**
- **DEVSECOPS.**

BSIDES
Ecuador

Red Teaming en Entornos Industriales

26 de junio del 2021

Cesar Cuadra
(Perú)

William Marchand
(Perú)

Las noticias del día

Supply chain

- Solarigate
  - Código fuente troyanizado
- Kasaya
  - “Ransomware as a Service”
- PyPi
  - 30 mil descargas de 8 paquetes
- Fortinet
  - CVE-2022-40684 AuthN Bypass

Los atacantes siguen ganando a los defensores

- Gobiernos : inteligencia accionable
- Delincuentes comúnes : dinero local

Cadenas de vulnerabilidades

- ProxyLogon
- ProxyShell
- ProxiNotShell

Escalar al máximo, minimizar el costo

- No buscar el 0day, robarlo al investigador

Existen aplicaciones en todos lados

- IoT
- IaC

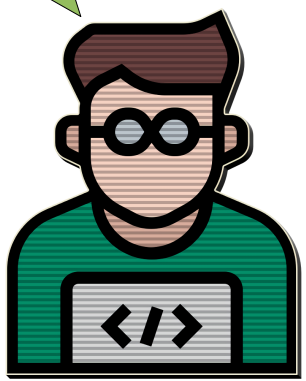
Security Breakers

Nu

DevSecOps

Recuerdan en la intro ?

- Apps libres de “bugs”
- Construir-Desplegar-Iterar
- Mantenerse con los cambios
- No hay tiempo para la seguridad

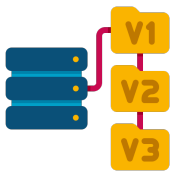


- Apps libres de vulnerabilidades
- Probar y Asegurar
- Mantenerse con las Amenazas
- No hay tiempo para los desarrolladores

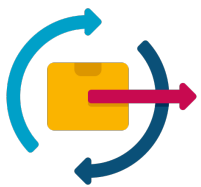
Aplicaciones Modernas

Secuencia DevOps Clásica (“pipeline”)

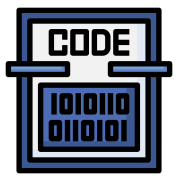
Desarrollo
- Spring
- Laravel
- React
- Express



Repositorio de Código Fuente
- Github
- Gitlab
- Bitbucket
- Atlassian Stash



CI/CD
- Jenkins



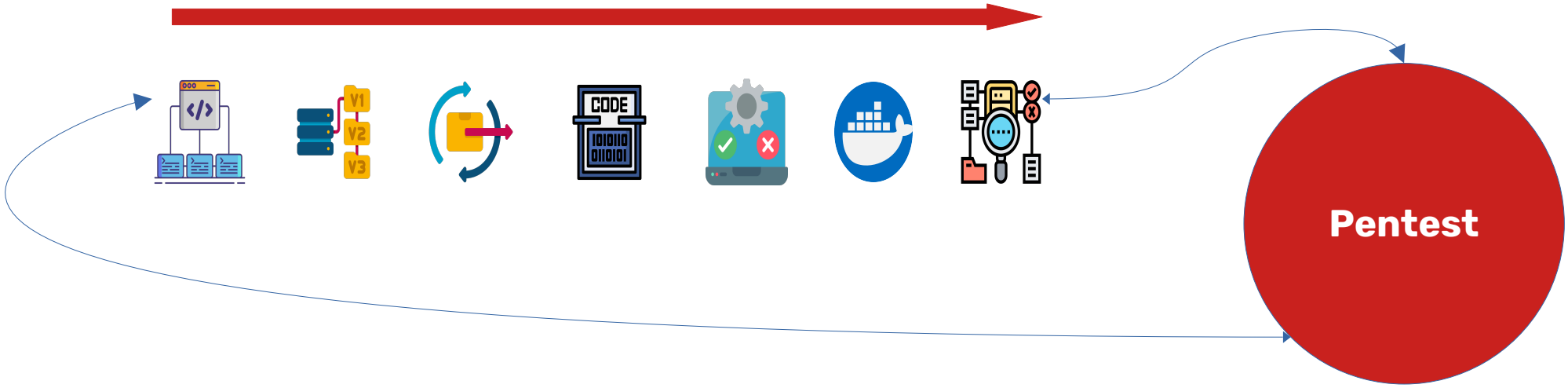
Repositorio de Binarios
- Jfrog
- Bintray
- Sonatype Nexus

QA/Staging/UAT
- Docker
- Kubernetes
- Tomcat
- MySQL

Producción
- Docker
- Kubernetes
- Tomcat
- MySQL

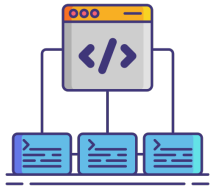
Monitoreo
- Zabbix
- Splunk
- ELK

Secuencia DevOps –> Pentest Clásica



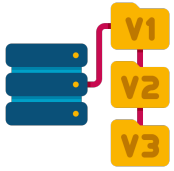
- Fallas en AuthN
- Fallas en AuthZ
- Fallas en reglas de negocio
- Todas las que un WAF no detiene

Aterrizando DevSecOps



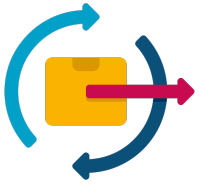
Desarrollo

Validaciones previas al commit



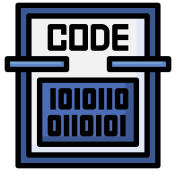
Repositorio de Código Fuente

Manejo de Secretos



CI/CD

- SAST
- SCA (Software Composition Analysis)



Repositorio de Binarios

- Ansible ???

QA/Staging/UAT

- DAST
- Gestión de Vulnerabilidades

Producción

- CaaC (Compliance as a Code)
- Seguridad en IaaC
- Análisis de Vulnerabilidades
- Pentesting

Monitoreo

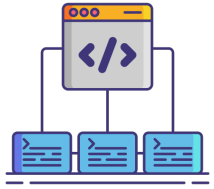
Enfocar en OWASP Top 10

Validaciones antes del Commit

- Talisman
  - https://github.com/thoughtworks/talisman
- Buscar “secretos” que pueden terminar en leaks
- Otras herramientas :
  - Crass - https://github.com/floyd-fuh/crass
  - Git Secret - https://git-secret.io/
  - Pre Commit - https://pre-commit.com/
  - Detect Secrets - https://github.com/Yelp/detect-secrets
  - Truffle Hog - https://github.com/dxa4481/truffleHog

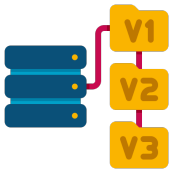
Validaciones antes del Commit

- Talisman
  - Instalar :
    - curl https://thoughtworks.github.io/talisman/install.sh > ~/install-talisman.sh
    - chmod +x ~/install-talisman.sh
  - Por ejemplo, para un solo proyecto :
    - cd pentester.one
    - ~/install-talisman.sh pre-commit
  - Validar en :
    - pentester.one/.git/hooks/pre-commit -> /home/user/.talisman/bin/talisman_hook_script
  - Ejecutará antes de cada `git commit`



Manejo de Secretos

- Usando repositorios de “secretos” para evitar que se “leakeen” en los repositorios de código fuente.
- Son sistemas que permiten manejar los “secretos” mediante servicios de encriptación basados en identidades
  - Secretos = API keys, passwords, certificados, etc.
  - Maneja autenticación y autorización.
- Hashicorp Vault - https://www.vaultproject.io/
  - El siguiente Lab muestra el uso básico de este
- Otros
  - Keywhiz - https://square.github.io/keywhiz/
  - EnvKey - https://www.envkey.com/
  - AWS Secrets Manager - https://aws.amazon.com/secrets-manager/

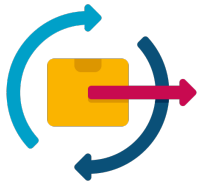


Lab : Manejo de secretos

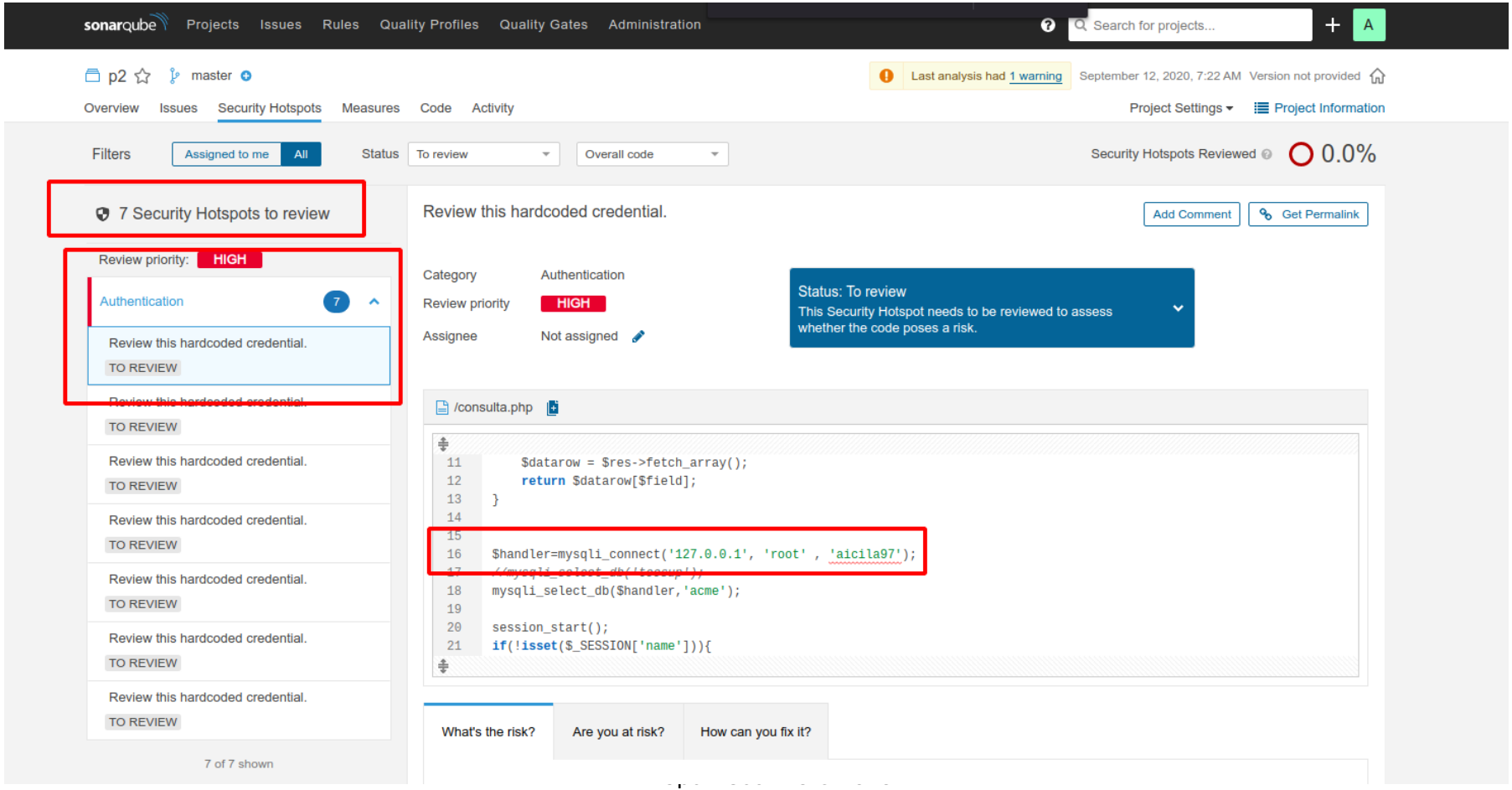
- Extraído de la página web de Vault
  - docker run -p 8200:8200 -e 'VAULT_DEV_ROOT_TOKEN_ID=dev-only-token' vault
  - pip install hvac
  - Usar el programa que esta en https://github.com/Open-Sec/Open-SecTraining/blob/master/milei.py
  - Buenas recomendaciones para ambientes productivos y, sobre todo, como manejar bien el asunto del token (no como root como en este Lab), se encuentra en https://developer.hashicorp.com/vault/tutorials/operations/production-hardening

SAST

- Lista de OWASP
  - https://owasp.org/www-community/Source_Code_Analysis_Tools
  - Destaquemos
    - FindBugs
    - FindSecBugs
    - Graudit
    - RIPS
    - SonarQube



SAST : Sonarqube



Lab : SAST : RIPS (PHP)

path / file: /var/www/html/acme/ subdirs
verbosity level: 1. user tainted only vuln type: All scan
code style: ayti bottom-up /regex/: search
windows
files user input
stats functions
RIPS 0.55

File: /var/www/html/acme/login.php

SQL Injection

Userinput reaches sensitive sink. For more information, press the help icon on the left side.

```
31: mysqli_query $result = mysqli_query($handler, $qry) or die ('<pre>' . mysqli_error($handler) . '</pre>');
    28: $qry = "SELECT * FROM `unica` WHERE usuario='$user' AND password='$pass';";
        • 24: $user = $_POST['username'];
        • 25: $pass = $_POST['password'];

    requires:
        18: if(isset($_POST['Login']))
```

hide all

File: /var/www/html/acme/consulta.php

SQL Injection

Userinput reaches sensitive sink. For more information, press the help icon on the left side.

```
42: mysqli_query $result = mysqli_query($handler, $getid) or die ('<pre>' . mysqli_error($handler) . '</pre>');
    41: $getid = "SELECT nombre, email FROM unica WHERE id = '$id'";
        • 39: $id = $_POST['id'];

    requires:
        35: if(isset($_POST['Submit']))
```

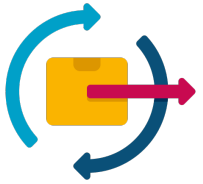
Cross-Site Scripting

Userinput reaches sensitive sink. For more information, press the help icon on the left side.

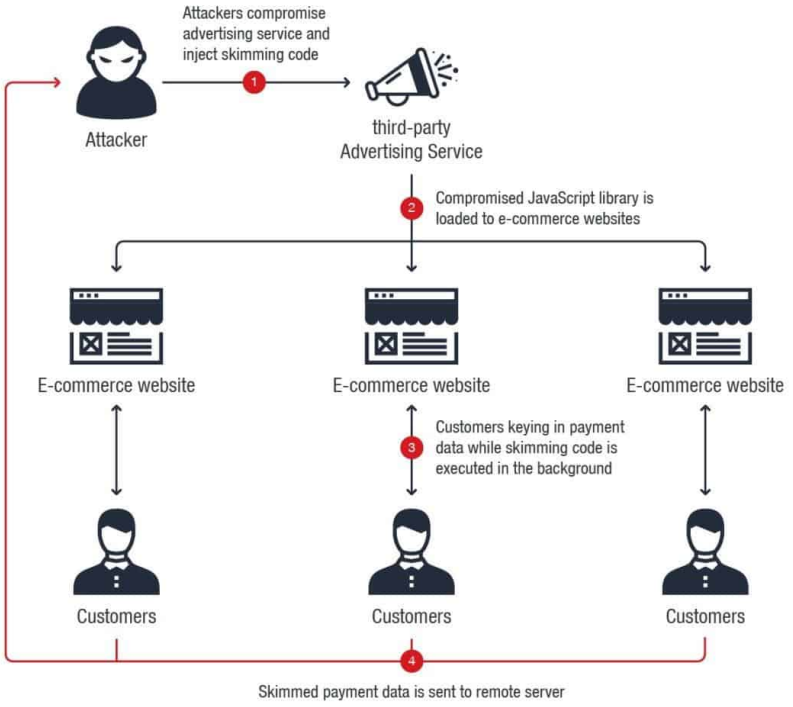
```
61: echo echo "<div class=\"body_padded\" align=\"center\">                <br />          <p> <img src=\"http://acme.open-sec.com/acme/oseh2015-
polo.png\"  width=\"256\" height=\"256\"> </p>          <br />        <h1>Bienvenido a las consultas del Admin</h1>          <div class=\"vulnerable_code_area\">                <h3>ID del Usuario:
</h3>                <form action=\"#\" method=\"POST\">                    <input type=\"text\" name=\"id\" size=\"4\" maxlength=\"4\">                <input type=\"submit\" name=\"Submit\" value=\"Consultar\">                </form
</div> </div> ";
    55: $html .= '</pre>';  // if(isset($_POST)),
        54: $html .= 'ID: ' . $id . '<br>Nombre: ' . $first . '<br>E-mail: ' . $last;  // if(isset($_POST)),
            53: $html .= '<pre>';  // if(isset($_POST)),
                33: $html = null;
            33: $html = null;
        • 39: $id = $_POST['id'];  // if(isset($_POST)),
            50: $first = mysqli_result ($result, $i, "nombre");  // if(isset($_POST)),
```

Software Composition Analysis (SCA)

- Recuerdan la intro y PyPi ?
- Otra forma ->

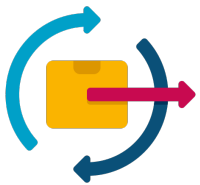


Imágen de TrendMicro



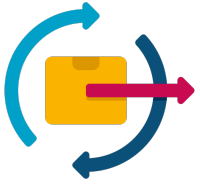
Software Composition Analysis (SCA)

- En simple : Análisis de dependencias y vulnerabilidades en las mismas
- Existen varias tools y algunas solamente trabajan con ciertos lenguajes
  - OWASP Dependency Check - https://www.owasp.org/index.php/OWASP_Dependency_Check
  - Retire JS - https://github.com/RetireJS/retire.js
  - Basadas en APIs y requieren registro
    - SonaType (Free for Open Source) - https://ossindex.sonatype.org/
    - Snyk (Free for Open Source) - https://snyk.io/



Lab SCA : dependency-check

- Crear un directorio donde descargar dependency-check
  - wget https://github.com/jeremylong/DependencyCheck/releases/download/v7.2.0/dependency-check-7.2.0-release.zip
  - El siguiente es un proyecto Java común y corriente
    - git clone https://github.com/ViniJP/GET-Java
  - Puede probar (si el tiempo lo permite) con otro en .Net
    - git clone https://github.com/dotnet-architecture/eShopOnWeb.git
  - Ejecutar el test
    - ./dependency-check.sh --project "GET" --scan "/home/user/.../GET-Java/"
    - ./dependency-check.sh --project "GET" --scan "/home/user/.../eShopOnWeb/"
  - El reporte en HTML (default) se encuentra en el mismo directorio
    - Usar la opción -o para indicar una ruta/nombre de reporte diferente



Lab SCA : dependency-check

DEPENDENCY-CHECK

Dependency-Check is an open source tool performing a best effort analysis of 3rd party dependencies; false positives and false negatives may exist in the analysis performed by the tool. Use of the tool and the reporting provided constitutes acceptance for use in or OWASP be held liable for any damages whatsoever arising out of or in connection with the use of this tool, the analysis performed, or the resulting report.

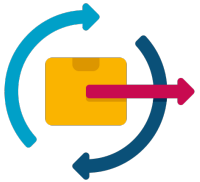
How to read the report | Suppressing false positives | Getting Help: github issues

Sponsor

Project: GET

Scan Information (show all):

- *dependency-check version*: 7.2.0
- *Report Generated On*: Wed, 19 Oct 2022 10:01:33 -0500
- *Dependencies Scanned*: 1 (1 unique)
- *Vulnerable Dependencies*: 1
- *Vulnerabilities Found*: 7
- *Vulnerabilities Suppressed*: 0
- ...



Summary

Display: Showing Vulnerable Dependencies (click to show all)

Dependency	Vulnerability IDs	Package	Highest Severity	CVE Count	Confidence	Evidence Count
gradle-wrapper.jar	cpe:2.3:a:gradle:gradle:4.6:*:*:*:*:*:*:*		CRITICAL	7	High	8

Dependencies

gradle-wrapper.jar

File Path: /home/wcuestas/eko2022/OSAH/devsecops/GET-Java/gradle/wrapper/gradle-wrapper.jar
MD5: 451e0b3037c608b724985f74784e7bb7
SHA1: 636cf935a0fd1451657a4112974b3500cce3ab84
SHA256:381dff8aa434499aa93bc25572b049c8c586a67faff2c02f375e4f23e17e49de

Evidence

DAST

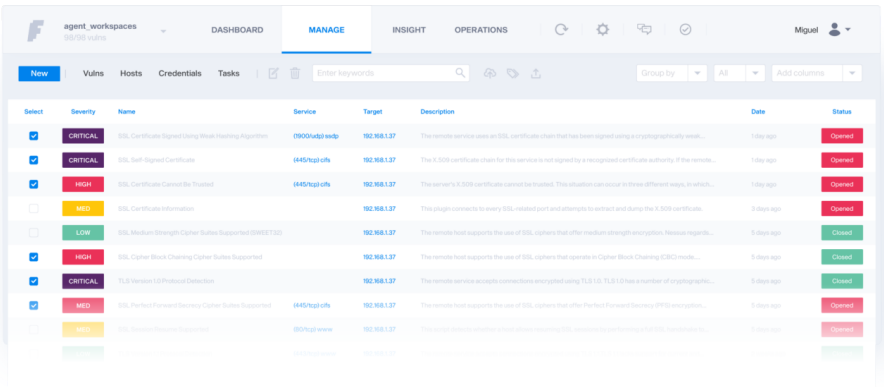
- Es lo que hemos venido haciendo estos días
  - Burp is the king in town
    - La parte de escaneo es bien útil para ciertas cosas, pero, manejado por endpoints específicos
    - La mejor forma de integrarlo en el pipeline es a través de su API Restful
  - ZAP sigue siendo bueno
- Es fácil confundir DAST con “escaneo automatizado”
  - Podríamos decir que lo que se ha venido haciendo en el curso es un “DAST manual” con un “DAST automatizado” en algunos puntos
  - El “DAST manual” es más para un pentest y el “DAST automatizado” sirve para el DevSecOps pipeline y el pentest

DAST

- Un DAST que se pueda incrustar en el pipeline de forma automatizada, también, es Nuclei
- Un caso de uso es mediante GitHub Actions
  - https://github.com/projectdiscovery/nuclei-action
- Como un plugin de Jenkins
  - https://plugins.jenkins.io/nuclei/

Gestión de Vulnerabilidades

- Herramientas que permitan Gestionar la Superficie de Ataque (ASM)
- Existen muchas tools hoy en día, pero, en DevSecOps consideraremos aquellas que permiten ejecutar y/o integrar los resultados de tools especializadas
- Algunos ejemplos
  - Faraday - https://github.com/infobyte/faraday/
  - ArcherySec - https://github.com/archerysec/archerysec
  - DefectDojo - https://www.defectdojo.org/



Compliance as a Code (CaaC)

- Dado que los ambientes en DevOps no se mantienen si no que se reconstruyen, el poder realizar test de cumplimiento con estándares que especifican requisitos técnicos resulta importante
- Estándares como PCI DSS, HIPAA, HITRUST, SOX, SWIFT son ejemplo de normativas que requieren ser contempladas dependiendo del tipo de organización

Compliance as a Code (CaaC)

- Casi cualquier escaneador de vulnerabilidades tiene opciones para validar el cumplimiento de normativas.
  - El reto es que se puedan personalizar e integrar en el pipeline
- Una buena tool es Inspec
  - https://www.inspec.io/

Lab CaaC : Validando cumplimiento

- Descargar Inspec desde https://www.chef.io/downloads/tools/inspec?os=debian
- Instalar
  - sudo dpkg –install inspec_5.18.14-1_amd64.deb
- Crear un directorio de trabajo como
  - inspec/controls
  - Dentro de controls se crearán las reglas de validación

Lab CaaC : Validando cumplimiento

File: **iptables.rb**

```ruby
control "xccdf_org.cisecurity.benchmarks_rule_3.6.1_Ensure_iptables_is_ Installed" do
    title "Ensure iptables is installed"
    desc "iptables allows configuration of the IPv4 tables in the linux kernel and the rules stored within them. Rationale: iptables is requiredfor fi
ewall management and configuration."
    impact 1.0
    tag "cis-rhel7-2.1.1":  "3.6.1"
    tag "level":  "1"
    tag "type": [  "Server",  "Workstation"]
    describe package('iptables')  do
        it { should be_installed }
    end
end
```

File: **nginx.rb**

```ruby
describe package('nginx') do
   it { should be_installed }
end
```

File: **tmp.rb**

```ruby
describe file('/tmp') do
   it { should be_directory }
end
```

El instructor proveerá los archivos fuente

Lab CaaC : Validando cumplimiento

```
[redacted]/inspec/controls$ inspec exec .

Profile:   tests from . (tests from .)
Version:   (not specified)
Target:    local://
Target ID: 77bd5bd0-cfd7-54ca-a102-0376b31abce5

  ✔  xccdf_org.cisecurity.benchmarks_rule_3.6.1_Ensure_iptables_is_ Installed: Ensure iptables is installed
     ✔  System Package iptables is expected to be installed

  System Package nginx
     ✔  is expected to be installed
  File /tmp
     ✔  is expected to be directory

Profile Summary: 1 successful control, 0 control failures, 0 controls skipped
Test Summary: 3 successful, 0 failures, 0 skipped
```

Ejecución

Lab CaaC : Validando cumplimiento

- Por ejemplo, el Requerimiento 6 de PCI DSS indica que se deben mantener seguros los sistemas y aplicaciones.
  - Una forma es reduciendo la superficie de ataque y establecer controles a nivel de red en los sistemas operativos como, por ejemplo, tener instalado y operativo un firewall
  - En la siguiente imagen se puede apreciar uno de los controles que valida que iptables tenga una regla DENY como default

Lab CaaC : Validando cumplimiento

File: **iptables-deny.rb**

```
        control  "cisecurity.benchmarks_rule_3.6.2_Ensure_default_deny_firewall_policy" do
    title "Ensure default deny firewall policy"
        desc "A default deny all policy on connections ensures that anyunconfigured network usage will be rejected. Rationale: With a    defaultaccept
policy the firewall will accept any packet that is not configured to be denied. It is easier to whitelist acceptable usage than to blacklist unaccepta
le usage."
    impact 1.0
    tag "cis-rhel7-2.1.1":  "3.6.2"
    tag "level":  "1"
    tag "type": [  "Server",  "Workstation"]
    %w[INPUT OUTPUT FORWARD].each do |chain|
        describe.one do
            describe iptables do
                it { should have_rule("-P #{chain} DROP") }
            end
            describe iptables do
                it { should have_rule("-P #{chain} REJECT") }
            end
        end
    end
end
```

Lab CaaC : Validando cumplimiento

```
[redacted]/inspec/controls$ inspec exec iptables-deny.rb

Profile:   tests from iptables-deny.rb (tests from iptables-deny.rb)
Version:   (not specified)
Target:    local://
Target ID: 77bd5bd0-cfd7-54ca-a102-0376b31abce5

  ×  cisecurity.benchmarks_rule_3.6.2_Ensure_default_deny_firewall_policy: Ensure default deny firewall policy (6 failed)
     ×  Iptables is expected to have rule "-P INPUT DROP"
     expected Iptables to have rule "-P INPUT DROP"
     ×  Iptables is expected to have rule "-P INPUT REJECT"
     expected Iptables to have rule "-P INPUT REJECT"
     ×  Iptables is expected to have rule "-P OUTPUT DROP"
     expected Iptables to have rule "-P OUTPUT DROP"
     ×  Iptables is expected to have rule "-P OUTPUT REJECT"
     expected Iptables to have rule "-P OUTPUT REJECT"
     ×  Iptables is expected to have rule "-P FORWARD DROP"
     expected Iptables to have rule "-P FORWARD DROP"
     ×  Iptables is expected to have rule "-P FORWARD REJECT"
     expected Iptables to have rule "-P FORWARD REJECT"


Profile Summary: 0 successful controls, 1 control failure, 0 controls skipped
Test Summary: 0 successful, 6 failures, 0 skipped
```

Ejecución

Lab CaaC : Validando cumplimiento

```
/inspec/controls$ inspec exec . -t ssh://hera@d                    --password=

Profile:   tests from . (tests from .)
Version:   (not specified)
Target:    ssh://hera@d
Target ID: e40b40e4-3047-5d09-ac1f-67283134aea9

  ×  cisecurity.benchmarks_rule_3.6.2_Ensure_default_deny_firewall_policy: Ensure default deny firewall policy (6 failed)
     ×  Iptables is expected to have rule "-P INPUT DROP"
     expected Iptables to have rule "-P INPUT DROP"
     ×  Iptables is expected to have rule "-P INPUT REJECT"
     expected Iptables to have rule "-P INPUT REJECT"
     ×  Iptables is expected to have rule "-P OUTPUT DROP"
     expected Iptables to have rule "-P OUTPUT DROP"
     ×  Iptables is expected to have rule "-P OUTPUT REJECT"
     expected Iptables to have rule "-P OUTPUT REJECT"
     ×  Iptables is expected to have rule "-P FORWARD DROP"
     expected Iptables to have rule "-P FORWARD DROP"
     ×  Iptables is expected to have rule "-P FORWARD REJECT"
     expected Iptables to have rule "-P FORWARD REJECT"
  ✔  xccdf_org.cisecurity.benchmarks_rule_3.6.1_Ensure_iptables_is_ Installed: Ensure iptables is installed
     ✔  System Package iptables is expected to be installed

  System Package nginx
     ×  is expected to be installed
     expected that `System Package nginx` is installed
  File /tmp
     ✔  is expected to be directory

Profile Summary: 1 successful control, 1 control failure, 0 controls skipped
```

MEJOR USAR SSH KEYS

Ejecución remota

Infrastructure as a Code (IaaC)

- Aún confían en el Docker Hub ?
  - No hay un compromiso de validación que no haya malware en esas imágenes
  - Lo mejor es escanear las imágenes antes de correrlas
- Anchore Engine
  - https://github.com/anchore/anchore-engine
- Otras tools
  - Clair - https://github.com/coreos/clair
  - Dagda - https://github.com/eliasgranderubio/dagda

Lab IaaC : Anchore-Engine

- Todo esta basado en contenedores
- Instalación
  - curl -O https://engine.anchore.io/docs/quickstart/docker-compose.yaml
  - docker-compose up -d
- Confirmar que los servicios de Anchore estan ejecutando (API y tools)
  - docker-compose ps

```
[redacted]$ docker-compose ps
/usr/lib/python3/dist-packages/paramiko/transport.py:219: CryptographyDeprecationWarning: Blowfish has been deprecated
  "class": algorithms.Blowfish,
         Name                       Command                  State                                  Ports
-------------------------------------------------------------------------------------------------------------------------
devsecops_analyzer_1        /docker-entrypoint.sh anch ...   Up (health: starting)   8228/tcp
devsecops_api_1             /docker-entrypoint.sh anch ...   Up (health: starting)   0.0.0.0:8228->8228/tcp,:::8228->8228/tcp
devsecops_catalog_1         /docker-entrypoint.sh anch ...   Up (health: starting)   8228/tcp
devsecops_db_1              docker-entrypoint.sh postgres    Up (health: starting)   5432/tcp
devsecops_policy-engine_1   /docker-entrypoint.sh anch ...   Up (health: starting)   8228/tcp
devsecops_queue_1           /docker-entrypoint.sh anch ...   Up (health: starting)   8228/tcp
```

Lab IaaC : Anchore-Engine

- Escanear imágenes
  - Validar que los “feeds” estén actualizados
    - docker-compose exec api anchore-cli system feeds list
  - Pedir a Anchore que descargue una imagen y la escanee
    - docker-compose exec api anchore-cli image add docker.io/library/debian:7
  - Revisar el estado del escaneo
    - docker-compose exec api anchore-cli image wait docker.io/library/debian:7

Lab IaaC : Anchore-Engine

```
[redacted]$ docker-compose exec api anchore-cli image wait docker.io/library/debian:7
/usr/lib/python3/dist-packages/paramiko/transport.py:219: CryptographyDeprecationWarning: Blowfish has been deprecated
  "class": algorithms.Blowfish,
Image Digest: sha256:81e88820a7759038ffa61cff59dfcc12d3772c3a2e75b7cfe963c952da2ad264
Parent Digest: sha256:2259b099d947443e44bbd1c94967c785361af8fd22df48a08a3942e2d5630849
Analysis Status: analyzed
Image Type: docker
Analyzed At: 2022-10-19T20:01:42Z
Image ID: 10fcec6d95c4a29f49fa388ed39cded37e63a1532a081ae2386193942fc12e21
Dockerfile Mode: Guessed
Distro: debian
Distro Version: 7
Size: 100884480
Architecture: amd64
Layer Count: 1

Full Tag: docker.io/library/debian:7
Tag Detected At: 2022-10-19T20:01:13Z
```

Status del escaneo (finalizado)

Lab IaaC : Anchore-Engine

- Listar contenidos de la imagen (os = operating system)
  - docker-compose exec api anchore-cli image content docker.io/library/debian:7 os
- Listar vulnerabilidades encontradas
  - docker-compose exec api anchore-cli image vuln docker.io/library/debian:7 all

No menos importante : cómo hacer pentesting basado en SCRUM ?

Acciones Scrum	Controles de Seguridad
Planificación de Lanzamiento	Diseño de Seguridad de Alto Nivel
Planificación del Sprint	Asesoría Por Demanda
Programación (Ejecución)	Verficiación de Controles
“Congelamiento” del Código	Pruebas Automatizadas
Pruebas de Regresión	Pruebas Automatizadas
Lanzamiento	Pruebas de Penetración Externas