Servicio de detección temprana de vulnerabilidades basado en Shodan: caso de estudio ESPE-CERT

Reyes Narváez, Jorge Luis

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Sistemas e Informática

Trabajo de titulación, previo a la obtención del título de Ingeniero en Sistemas e Informática

Dr. Fuertes Díaz, Walter Marcelo, PhD.

22 de noviembre del 2021

Informe de originalidad

NOMBRE DEL CURSO
Revisión de tesis

NOMBRE DEL ALUMNO
JORGE LUIS REYES NARVAEZ

NOMBRE DEL ARCHIVO
JORGE LUIS REYES NARVAEZ - Documento sin título

SE HA CREADO EL INFORME
22 nov 2021

Resumen

Fragmentos marcados	15	2 %
Fragmentos citados o entrecomillados	3	0,3 %

Coincidencias de la Web

wikipedia.org	3	0,4 %
welivesecurity.com	2	0,3 %
mitre.org	1	0,2 %
freejournal.info	1	0,2 %
synology.cn	1	0,2 %
europa.eu	2	0,2 %
elsevier.com	2	0,2 %
semanticscholar.org	1	0,1 %
cvedetails.com	1	0,1 %
ieee.org	1	0,1 %
orcid.org	1	0,1 %
uni-trier.de	1	0,1 %
sciencedirect.com	1	0,1 %

1 de 18 fragmentos
Fragmento del alumno CITADO

firmado electrónicamente por:
WALTER MARCELO
FUERTES DIAZ

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Sistemas e Informática

Certificación

Certifico que el trabajo de titulación: "**Servicio de detección temprana de vulnerabilidades basado en Shodan: caso de estudio ESPE-CERT**" fue realizado por el señor **Reyes Narváez, Jorge Luis** el cual ha sido revisado y analizado en su totalidad por la herramienta de verificación de similitud de contenido. Además cumple con los requisitos legales, teóricos, científicos, técnicos y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, razón por la cual me permito acreditar y autorizar para que lo sustente públicamente.

Sangolquí, 22 de noviembre de 2021

Firmado electrónicamente por:
WALTER MARCELO
FUERTES DIAZ

Ing. Walter Marcelo Fuertes Díaz, PhD.

C.C. 1707017701

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Sistemas e Informática

Responsabilidad de Autoría

Yo, **Reyes Narváez, Jorge Luis**, con cédula de ciudadanía n° 1723002836, declaro que el contenido, ideas y criterios del trabajo de titulación: **Servicio de detección temprana de vulnerabilidades basado en Shodan: caso de estudio ESPE-CERT** es de mi autoría y responsabilidad, cumpliendo con los requisitos legales, teóricos, científicos, técnicos, y metodológicos establecidos por la Universidad de las Fuerzas Armadas ESPE, respetando los derechos intelectuales de terceros y referenciando las citas bibliográficas.

Sangolquí, 22 de noviembre de 2021

Reyes Narváez, Jorge Luis

C.C. 1723002836

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

Departamento de Ciencias de la Computación

Carrera de Ingeniería de Sistemas e Informática

Autorización de Publicación

Yo, **Reyes Narváez, Jorge Luis**, con cédula de ciudadanía n° 1723002836, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar el trabajo de titulación: **Servicio de detección temprana de vulnerabilidades basado en Shodan: caso de estudio ESPE-CERT** en el Repositorio Institucional, cuyo contenido, ideas y criterios son de mi responsabilidad.

Sangolquí, 22 de noviembre de 2021

Reyes Narváez, Jorge Luis
C.C. 1723002836

Dedicatoria

Dedico el presente trabajo de investigación a los pilares fundamentales en mi vida, a mis padres y hermana, quienes me han guiado en cada decisión con sus consejos y palabras de aliento, pero sobretodo con su sabiduría y paciencia. Esto es para ustedes y por ustedes, se lo merecen.

Jorge Reyes

Agradecimientos

Agradezco primero a Dios por la darme la vida, la inteligencia y la fortaleza necesaria para enfrentarme a cada reto.

Agradezco a mis padres y a mi hermana, por su apoyo incondicional en cada obstáculo, por enseñarme que el esfuerzo tiene las mejores recompensas y sobretodo por enseñarme a ser paciente.

Agradezco a mi director de tesis, por la confianza brindada para materializar mis ideas, por guiarme y apoyarme en cada fase de este proceso.

Jorge Reyes

Tabla de Contenidos

Certificación 3

Responsabilidad de Autoría 4

Autorización de Publicación 5

Dedicatoria 6

Agradecimientos 7

Resumen 14

Abstract 15

Capítulo I 16

Aspectos Generales 16

- Antecedentes 16
- Problemática 17
- Justificación 20
- Objetivos 21
  - Objetivo General 21
  - Objetivos Específicos 22
- Alcance 22

Capítulo II 23

Marco Teórico y Conceptual 23

- Marco Conceptual 23
  - Equipo de Respuesta ante Emergencias Informáticas (CERT) 23
  - Definición 23
  - Tipos de CERT 24
  - Servicios de un CERT 25
  - Vulnerabilidades de Seguridad 26
  - Vulnerabilidades de hardware 27

Vulnerabilidades de software 27
CVE (Common Vulnerabilities and Exposures) 28
NVD (National Vulnerability Database) 28
Marco Teórico 29
Sistemas de Escaneo y Detección Temprana de Vulnerabilidades 29
Priorización de vulnerabilidades 49
Capítulo III 54
Marco Metodológico 54
Metodología y Proceso de desarrollo para los VDS 54
Propuesta de metodología de detección de vulnerabilidades 54
Propuesta de proceso de desarrollo del VDS 55
Metodología multifactorial de recolección, análisis y toma de decisiones 56
OSINT (Open Source Intelligence) 57
Fase de Colección 58
Shodan 58
Fase de análisis 62
Variables Globales 62
Variables de IP 66
Variables de Vulnerabilidad 69
Proceso de extracción de conocimientos 72
Capítulo IV 76
Desarrollo del prototipo de detección y priorización de vulnerabilidades 76
Definición del enfoque de detección de vulnerabilidades 77
Diseño Arquitectura del sistema 77
Definición la metodología de desarrollo 80
Modelo de prototipos 80
Desarrollo del prototipo 82
Validación de la Operación del Prototipo 85
Módulo Dashboard General de Vulnerabilidades 86

Banners Detallados de Vulnerabilidades por cada IP 90
Módulo de tabla de priorización 93
Análisis de priorización 93
Capítulo V 99
Conclusiones y Recomendaciones 99
Conclusiones 99
Recomendaciones 101
Referencias 103

Índice de Tablas

Tabla 1 Vulnerabilidades de Software 34

Tabla 2 Vulnerabilidades de IOT y Red 39

Tabla 3 Vulnerabilidades en aplicaciones y servicios web 42

Tabla 4 Variables de especificación del banner de Shodan 59

Tabla 5 Correspondencia entre la puntuación CVSS y el valor cualitativo (gravedad) 64

Tabla 6 Variables del enfoque de detección de vulnerabilidades 77

Tabla 7 IP para el proceso de escaneo 86

Índice de Figuras

Figura 1 Artículos publicados por base de datos indexada 31

Figura 2 Distribución de los estudios preseleccionados y primarios a los VDAs 33

Figura 3 Tendencia del tiempo en la detección de vulnerabilidades 45

Figura 4 Principales parámetros para el desarrollo de los VDS 47

Figura 5 Fases de la metodología de detección de vulnerabilidades 55

Figura 6 Fases del proceso de desarrollo de VDSs 56

Figura 7 Fases OSINT 58

Figura 8 Número de datos y vulnerabilidades cuando se muestra la etiqueta de consulta 69

Figura 9 Factor de riesgo 74

Figura 10 Diagrama de alto nivel que muestra el flujo de datos en la plataforma Shodan 78

Figura 11 Arquitectura del consumo de recursos del prototipo y diagrama de procesamiento de datos 80

Figura 12 Modelo de Prototipos 82

Figura 13 Información General de la Organización 87

Figura 14 Cantidad de Vulnerabilidades por IP 87

Figura 15 Cantidad de Puertos Abiertos 88

Figura 16 Años de referencia en el identificador CVE-ID 88

Figura 17 Cantidad de Query Tags 89

Figura 18 Visión General de CVSS Score del total de vulnerabilidades 90

Figura 19 Banner de información detallada por cada IP 91

Figura 20 Enlace a los hostnames detectados 91

Figura 21 Enlace a los puertos detectados 92

Figura 22 Enlace a NVD 92

Figura 23 Tabla de priorización de vulnerabilidades 93

Figura 24 Variables de IP 95

Figura 25 Variables de vulnerabilidad 96

Figura 26 Tendencia del CVSS y factor de riesgo por IP 97

Figura 27 Tendencia del CVSS y Factor de Riesgo por vulnerabilidad 98

Índice de Algoritmos

Algoritmo 1 Extracción de vulnerabilidades 83

Algoritmo 2 Adquisición de data de Shodan 84

Algoritmo 3 Cálculo del factor de riesgo 85

Resumen

El objetivo de la presente investigación es desarrollar un prototipo de detección y priorización de vulnerabilidades basado en Shodan, a través de una arquitectura Cliente-Servidor para los servicios reactivos y proactivos del ESPE-CERT. En primer lugar, se revisó de forma sistemática los enfoques, técnicas y herramientas en la implementación de sistemas de detección y escaneo de vulnerabilidades, por medio de la guía metodológica de Barbara Kitchenham. Los resultados demostraron que no existe una metodología estándar o predominante para el proceso de desarrollo, por lo cual se propuso una metodología con cinco fases y un proceso de desarrollo con seis fases para los sistemas de detección de vulnerabilidades. En segundo lugar, a través de la metodología OSINT se aplicaron las fases de colección, análisis y extracción de conocimiento. Los datos de entrada se obtuvieron a través de las REST API de Shodan, luego se aplicó un proceso matemático con la información relevante sobre vulnerabilidades y su entorno para poder cuantificar y calcular el factor de riesgo de cada vulnerabildad, logrando un orden de priorización. En tercer lugar, se construyó un servicio web mediante el modelo de prototipos, que permita extraer, correlacionar y analizar la información. Los resultados muestran que Shodan tiene variables relevantes la cuales permiten evaluar y cuantificar la sobreexposición información de una organización. Además, se identificó que CVSS no es suficiente para priorizar vulnerabilidades, ya que los entornos donde se identifican tienen características diferentes. Finalmente este estudio aporta con metodologías, un modelo de priorización y un prototipo para los servicios del ESPE-CERT.

Palabras clave:

- **DETECCIÓN DE VULNERABILIDADES**
- **CIBERSEGURIDAD**
- **SHODAN**

Abstract

The objective of the present research is to develop a prototype of vulnerability detection and prioritization based on Shodan information, through a Client-Server architecture for ESPE-CERT's reactive and proactive services. First, approaches, techniques and tools in the implementation of vulnerability detection and scanning systems were systematically reviewed through Barbara Kitchenham's methodological guide. The results showed that there is no standard or predominant methodology for the development process, so a five-phase methodology and a six-phase development process for vulnerability detection systems were proposed. Secondly, through the OSINT methodology, the phases of knowledge collection, analysis and extraction were applied. The input data were obtained through Shodan's REST APIs, then a mathematical process was applied with the relevant information on vulnerabilities and their environment in order to quantify and calculate the risk factor of each vulnerability, achieving an order of prioritization. Thirdly, a web service was built using the prototype model to extract, correlate and analyze the information. The results show that Shodan has relevant variables which allow to evaluate and quantify the information overexposure of an organization. In addition, it was identified that CVSS is not sufficient to prioritize vulnerabilities, since the environments where they are identified have different characteristics. Finally, this study provides methodologies, a prioritization model and a prototype for ESPE-CERT's proactive and reactive services.

Keywords:

- **VULNERABILITY DETECTION**
- **CYBERSECURITY**
- **SHODAN**

Capítulo I

Aspectos Generales

Antecedentes

El constante crecimiento de Internet ha generado en las empresas, instituciones educativas e infraestructuras críticas una dependencia casi total, donde uno de los principales objetivos es brinar a cada uno de los ciudadanos la posibilidad de acceder a los servicios a través de Internet, por esta razón la confidencialidad, integridad, y disponibilidad de la información se han vuelto los pilares fundamentales en su funcionamiento diario (Heide, 2017).

La ciberseguridad se define como la práctica para prevenir daños, proteger y restaurar sistemas, redes o programas de ataques digitales (Cisco, 2021) (NIST, 2021). Con el rápido avance tecnológico y con el afán de brindar mejores servicios y soluciones a sus clientes, las organizaciones han sido más dependientes de su conectividad digital, motivo por el cual existe la necesidad de invertir en personal calificado y herramientas que permitan establecer y aplicar medidas de ciberseguridad que resguarden la información y protejan las infraestructuras.

Desde el 2013 hasta la actualidad las grandes empresas, PYMES, instituciones educativas, gobiernos e infraestructuras críticas se han visto afectadas por la era del ransomware, el cual aprovecha vulnerabilidades para inhabilitar los activos informáticos, de esta forma los servicios y los datos quedan inutilizables al obligar en muchos casos a pagar millones de dólares por su rescate (SOPHOS, 2020). El prestigio y reputación de las organizaciones es directamente proporcional con el tiempo que tardan en restablecer los servicios, al llegar a generar grandes pérdidas económicas, que en muchos casos es irreparable y obliga sobre todo a las PYMES a finalizar sus actividades comerciales debido a que el impacto económico es mayor.

El Observatorio de Ciberseguridad en América Latina y el Caribe ha trabajado desde el 2016 en fortalecer las capacidades de la región en materia de ciberseguridad, de acuerdo con un estudio realizado, el nivel de madurez promedio de la región está entre 1 y 2, de acuerdo con el Modelo de Madurez de la Capacidad de Ciberseguridad para las Naciones (CMM) (en el que 1 significa etapa Inicial y 5 significa Dinámica o Avanzada), los gobiernos e instituciones han mostrado interés y cooperación para fortalecer sus conocimientos y capacidades, las cuales les permitan establecer mecanismos de monitoreo, análisis y evaluaciones de impacto relacionados con la ciberseguridad (OEA, Observatorio de la Ciberseguridad en América Latina y El Caribe, n.d.).

La clave para que las organizaciones no sean sorprendidas por los delincuentes es la respuesta rápida, eficaz y coordinada por parte de los equipos de respuesta ante incidentes: CERT (Equipo Respuesta a Emergencias Informáticas), CSIRT (Equipo de Respuesta a Incidentes de Seguridad Informática), CIRT (Equipo de Respuesta a Incidentes Informáticos), IRT (Equipo de Respuesta a Incidentes), ERT (Equipo de Respuesta a Emergencias), SIRT (Equipo de Respuesta a Incidentes de Seguridad), SERT (Equipo de Respuesta a Emergencias de Seguridad), quienes deben estar preparados con personal calificado y herramientas adecuadas que les permitan mantener un monitoreo constante, establecer protocolos y procesos de seguridad, los cuales ayuden a mitigar el impacto y contraatacar el accionar de los delincuentes (Mena, y otros, 2018).

Problemática

La pandemia mundial del COVID-19 ha marcado un hito en la transformación digital en las organizaciones, el teletrabajo plantea nuevos retos ya que amplía el perímetro de seguridad hasta las redes domésticas. La protección de la información es un trabajo constante que requiere de mucho esfuerzo a nivel técnico y tecnológico,

debido a que representa un impacto negativo a nivel funcional y reputacional para las organizaciones el perderla o exponerla, en este contexto, la rápida transición sin duda ha dejado vulnerabilidades expuestas las cuales están siendo aprovechadas por los delincuentes (SOPHOS, 2020).

La Transformación Digital representa un desafío para las empresas, donde se debe considerar a la ciberseguridad como aspecto integral de cada uno de sus procesos. De no hacerlo, será cada vez más común ver problemas asociados a bases de datos mal configuradas (por ejemplo, filtraciones de información masivas por una mala configuración de servidores), y que la adopción de nuevas tecnologías que buscan brindar mayor seguridad genere más problemas que soluciones (cámaras, controles de acceso biométricos, entre otros) (OEA, Inter - American Development Bank, 2020).

De acuerdo con una encuesta realizada por (ESET, ESET, 2018) a instituciones de primaria, secundaria y universidades en Latinoamérica para conocer el nivel de exposición a riesgos de seguridad, el 67% de las instituciones educativas aseguró haber sufrido al menos un incidente de seguridad, además de acuerdo con un reporte de 2020 (ESET, Welivesecurity, 2020) en donde participaron casi 4000 empresas de diferentes tamaños, 1 de cada 3 en América Latina aseguró haber sido víctima de la ciberdelincuencia, dentro de los incidentes que las empresas e instituciones educativas han reportado están infección de malware, ransomware, explotación de vulnerabilidades, acceso indebido a aplicaciones, ataques de ingeniería social y denegación de servicio.

Los dispositivos que forman parte de las infraestructuras informáticas de las empresas e instituciones educativas pueden verse afectados por vulnerabilidades descubiertas en los procesos de investigación que mantienen varias empresas a nivel mundial sobre los diferentes productos de hardware o software, se estima que un 32%

de las vulnerabilidades se deben a malas configuraciones del personal encargado, y en un 52% las vulnerabilidades son descubiertas o generadas directamente por el accionar de los delincuentes, quienes intentan desestabilizar las organizaciones con el objetivo de obtener beneficios económicos. Alrededor de 86 millones de dólares es el costo promedio a nivel global de una brecha de datos y explotación de vulnerabilidades para una organización, donde el 80% incluye la fuga de información personal, mientras que en el 32% se compromete propiedad intelectual (IBM Corporation, 2020).

El ciberdelito en el Ecuador ha ido en aumento donde el uso generalizado de Internet y el ciberespacio han permitido el desarrollo de nuevas formas delictivas de carácter transnacional que dañan los intereses patrimoniales y personales de los usuarios (Ron M. , Fuertes, Bonilla, Toulkeridis, & Diaz, 2018), además es el segundo país con mayor porcentaje de incidentes de seguridad (ESET, Welivesecurity, 2020), los sectores de Educación (4.4%) y Profesional Científico, Técnico (8.4%) suman un total de 12.8% de ataques que se han registrado, esto quiere decir que de cada 100 ataques, aproximadamente 13 están dirigidos a estos sectores (Hackmageddon, 2020).

De acuerdo con estos datos la Universidad de las Fuerzas Armadas “ESPE” al ser una institución educativa y profesional no está exenta del accionar por parte de actores maliciosos, de hecho con base en los datos se encuentra entre los objetivos constantes por parte de los ciberdelincuentes, quienes aprovechan las vulnerabilidades para extraer información, secuestrar datos o comprometer los sistemas, por tal razón es importante mantener un monitoreo continuo a través de los servicios reactivos y proactivos del ESPE-CERT con el objetivo de reducir el tiempo de detección de vulnerabilidades. Se ha identificado que alrededor de 280 días es el tiempo promedio que tarda una organización a nivel global para identificar y contener

una vulnerabilidad, mientras que, en América Latina, el tiempo promedio es de 328 días, lo cual representa un lapso de tiempo considerable y por supuesto un riesgo inminente ya que la probabilidad de explotación es mayor (IBM Corporation, 2020).

Justificación

La capacidad de detección, priorizacion y respuesta a incidentes de seguridad informática implica una variedad de decisiones e innovaciones las cuales permiten organizar y sistematizar la información (Andrade, Cordova, Ortiz-Garcés, Fuertes, & Cazares, 2020). El Equipo de Respuesta para Emergencias Informáticas (CERT) o el Equipo de Respuesta a Incidentes de Seguridad Informática (CSIRT), son responsables de proporcionar servicios de detección y respuesta a incidentes de una organización, además recibe información sobre posibles incidentes, los previene de ser posible o de suceder los investiga y toma acción mediante políticas que permitan asegurar que los daños causados por los incidentes puedan ser minimizados (Hellwig, et al., 2016).

Los incidentes de seguridad informática tienen efectos fatales en las organizaciones, la desfiguración de páginas web, servicios fuera de línea, la exposición de información sensible, la infección de servidores, acceso no autorizado y el secuestro de datos, son solo algunos de los incidentes que podrían desestabilizar y desprestigiar totalmente las infraestructuras tecnológicas, por tal motivo la detección y mitigación de vulnerabilidades es vital para evitar que el cibercrimen se convierta en un hecho.

Las pérdidas económicas cada año van en aumento, las amenazas informáticas son preocupantes tanto en la región como en el mundo entero, existen estadísticas alarmantes de ataques a infraestructuras tanto públicas como privadas (Securelist, 2020). Ninguna organización está exenta de un ataque, motivo por el cual

tienen la responsabilidad de asegurar sus activos informáticos de manera óptima y eficiente, para entregar sus servicios con calidad y de manera oportuna, donde la disponibilidad, integridad y confidencialidad de información sea prioritario para el crecimiento de la institución.

La Universidad de las Fuerzas Armadas “ESPE” almacena información sensible de sus estudiantes, personal docente y administrativo, y ofrece varios servicios a través de Internet al público en general, lo cual aumenta los escenarios de riesgo de seguridad debido al creciente número de vulnerabilidades y sofisticación de los ciberataques. Debido a la valiosa información que se maneja y el impacto social que generaría desestabilizar la infraestructura académica, la universidad es un objetivo predominante para los ciberdelincuentes, por tal motivo es importante conocer las vulnerabilidades para prevenir la explotación por parte de actores maliciosos.

Por lo anteriormente expuesto, la presente investigación resulta útil y de gran relevancia porque se enfoca en investigar y diseñar un prototipo de detección y priorización de vulnerabilidades basado en el potente motor de búsqueda Shodan lo cual fomenta el crecimiento de los servicios reactivos y proactivos del ESPE-CERT, con el objetivo de conocer de manera óptima y oportuna las vulnerabilidades.

Objetivos

Objetivo General

Desarrollar un prototipo de detección y priorización de vulnerabilidades basado en la información de Shodan, a través de una arquitectura Cliente-Servidor para los servicios reactivos y proactivos del ESPE-CERT.

Objetivos Específicos

- Realizar una revisión sistemática de literatura sobre los procesos de desarrollo de sistemas de escaneo y detección de vulnerabilidades para definir metodologías y herramientas.
- Desarrollar un modelo matemático que permita priorizar las vulnerabilidades identificadas en Shodan con variables de entorno desde el punto de vista del atacante.
- Diseñar la arquitectura de consumo y tratamiendo de información, basada en la escalabilidad e integración en los servicios reactivos y proactivos para el ESPE-CERT.
- Desarrollar un prototipo basado en Rest API de Shodan que permita automatizar el proceso de identificación, correlación y priorización de las vulnerabilidades para el ESPE-CERT a través del modelo de desarrollo de prototipos.

Alcance

Desarrollar un prototipo de detección y priorización de vulnerabilidades para los servicios reactivos y proactivos del ESPE-CERT que permita el tratamiento automatizado de la información recopilada de Shodan relacionada con vulnerabilidades.

Capítulo II

Marco Teórico y Conceptual

Marco Conceptual

Equipo de Respuesta ante Emergencias Informáticas (CERT)

El rápido avance tecnológico al que las organizaciones se enfrentan, el continuo crecimiento de dispositivos y personas que se conectan a Internet, y la competitividad por ofrecer servicios que faciliten la interacción entre cliente – empresa, han generado procesos de desarrollo e integración de sistemas acelerados, donde las infraestructuras de TI en todo el mundo han tenido que enfrentar grandes retos a nivel de seguridad.

Los datos estadísticos relacionados con incidentes de seguridad cada vez son mas alarmantes, motivo por el cual las organizaciones se han centrado en la protección de sus activos informáticos y para solventar esta necesidad se han conformado los diferentes CERT, por sus siglas en ingles “Computer Emergency Response Team”.

Definición

De acuerdo con (ENISA, 2006) el termino CERT esta registrado en Estado Unidos por el CERT Coordination Center (CERT/CC), por lo cual existen varios nombres que se han usado para identificar al mismo tipo de equipos, Computer Security Incedent Response Team (CSIRT), Incident Response Team (IRT), Computer Incident Response Team (CIRT), Security Emergency Response Team (SERT).

Un CERT es un equipo de expertos en seguridad TI quienes estan en la capacidad de responder ante un incidente de seguridad, con los servicios y soporte necesario. Su trabajo esta enfocado en prevenir y minizar el riesgo de las organizaciones a traves de varios servicios, con el objetivo de identificar, alertar e informar vulnerabilidades presentes tanto en software como en hardware.

Tipos de CERT

Uno de los factores mas importantes al momento de conformar un CERT es su enfoque y entorno de desarrollo, para poder establecer servicios basados en las reglas de negocio que apoyen a los objetivos organizacionales, la calsificación de los CERT según (ENISA, 2006) es la siguiente:

- CERT Académico
- CERT Comercial
- CERT Gubernamental
- CERT Militar
- CERT Interno
- CERT de Soporte
- CERT Nacional
- CERT para PYMES

De acuerdo con FIRST por sus siglas en inglés (Forum for Incident Response and Security Teams), quien tiene como objetivo certificar varios CERT/CSIRT alrededor del mundo para fomentar la comunicación y cooperación ante incidentes de seguridad, muestra en su página oficial (FIRST, 2021) que en Ecuador existen siete miembros certificados en su comunidad, los cuales se mencionan a continuación:

- BLUE HAT CERT
- CERT Radical
- CSIRT Telconet
- CSIRT-CEDIA
- CSIRT-EPN
- EcuCERT
- MAINTLATAM CSIRT

Servicios de un CERT

Los servicios que ofrecen los diferentes CERT alrededor del mundo dependen del entorno empresarial hacia el cual esten enfocados, ya que el objetivo es cumplir con las metas que cada institución se haya planteado, de forma general según (ENISA, 2021) se han divido en 3 categorías, como se describen a continuación:

Servicios reactivos. Este tipo de servicios son los de mayor impacto y que definen como tal a un CERT, ya que estan enfocados en dar una respuesta inmediata a peticiones de asistencia, es decir tratar activamente los incidentes alertados ya sea por el personal humano o por los sistemas de monitoreo. Algunos de los sub-servicios son:

- Alertar y advertencias
- Tratamiento de incidentes
- Manejo de vulnerabilidades
- Manejo de artefactos

Es fundamental para el aprendizaje continuo de un CERT que se documenten los eventos a través de informes y análisis post mortem de un incidente de seguridad ya que son el eje central para dar un soporte técnico fiable y de calidad.

Servicios proactivos. Están diseñados para detectar, alertar y prevenir las vulnerabilidades antes de que exista un ataque o incidente de seguridad, con el fin de evitar impactos negativos sobre las organizaciones, uno de los principales objetivos que tiene esta categoría de servicios es compartir información con su comunidad para proteger los activos informáticos. Algunos de los sub-servicios son:

- Auditorias de seguridad
- Desarrollo de herramientas
- Detección de intrusiones
- Intercambio de inteligencia sobre amenazas
- Configuración y mantenimiento de la seguridad

Servicios de Gestión de la Calidad de la Seguridad. Se enfocan en la mejora continua de la seguridad en las organizaciones, esta categoría es resultado del aprendizaje e información recolectada de las categorías mencionadas anteriormente. Los resultados que se desean obtener son a largo plazo ya que se centran en el aprendeizaje y capacitacion, algunos de los sub-servicios son:

- Análisis de riesgos, continuidad del negocio y recuperación tras un desastre
- Sensibilización
- Consultoría de seguridad
- Evaluación o certificación de productos

Vulnerabilidades de Seguridad

Las vulnerabilidades son cualquier tipo de fallo ya sea de software o hardware que comprometa la integridad, disponibilidad o confidencialidad de la información,

siendo indispensable identificarlas y tratarlas en el menor tiempo posible (INCIBE, 2017).

Vulnerabilidades de hardware

Son aquellos fallos presentes a nivel físico en los componentes de hardware, por ejemplo, procesadores, memorias RAM, puertos USB, etc.

Vulnerabilidades de software

Son los errores existentes a nivel de programación, configuración y diseño relacionados con los sistemas operativos y las aplicaciones.

Todas las vulnerabilidades pueden ser explotadas a través de varias Tácticas, Técnicas y Procedimientos (TTP). Principalmente en las técnicas se identifica la forma en que un ciberdelincuente aprovecha las vulnerabilidades generalmente mediante el uso de Malware, que es un código encargado de aprovechar los fallos para lograr un objetivo ya sea de robar información, observar, conceder permisos, etc. A nivel de ciberseguridad las TTP marcan el camino que siguen los ciberdelincuentes para lograr la explotación de una vulnerabilidad.

Alrededor del mundo siempre va a existir una persona o grupo de personas que intenten comprometer las diferentes infraestructuras tecnológicas, este riesgo continuo al que se enfrentan los expertos en ciberseguridad en las organizaciones se conoce como Amenaza Persistente Avanzada (APT), que es un ataque de niveles sofisticados con el objetivo de filtrar información (NIST, 2012).

Las vulnerabilidades pueden ser identificadas por procesos de auditoría interna o a través mecanismos de monitoreo propios, pero además con el objetivo de combatir conjuntamente a los ciberdelincuentes se han creado plataformas públicas y de uso gratuito donde se documentan las vulnerabilidades conocidas, la CISA por sus

siglas en inglés (Cybersecurity and Infrastructure Security Agency) ha patrocinado programas que trabajan en conjunto para mantener informada a la comunidad acerca de las vulnerabilidades que las diferentes empresas proveedoras ya sea de software o hardware más representativas a nivel mundial han detectado en sus laboratorios de investigación.

Los programas son CVE por sus siglas en inglés (Common Vulnerabilities and Exposures) y NVD por sus siglas en inglés (National Vulnerability Database) los cuales se detallan a continuación:

CVE (Common Vulnerabilities and Exposures)

Es una lista de registros lanzada por MITRE Corporation en 1999, cada registro tiene un número de identificación, una descripción y al menos una referencia pública para las vulnerabilidades conocidas públicamente. Los registros CVE se utilizan en numerosos productos y servicios de ciberseguridad de todo el mundo, incluido NVD (CVE, 2020).

NVD (National Vulnerability Database)

Es una base de datos de vulnerabilidades que esta sincronizada con la lista de registros CVE, en caso de existir actualizaciones de CVE se visualizara inmediatamente en NVD (CVE, 2020).

De forma general CVE alimenta a NVD, esta plataforma se basa en la información que contienen los registros CVE, y aumentan ciertos parámetros como información de solución, puntajes de gravedad a través de CVSS por sus siglas en inglés (Common Vulnerability Scoring System) y calificaciones de impacto. Adicionalmente NVD proporciona funciones de búsqueda avanzadas; por sistema operativo; por nombre de proveedor, nombre de producto y / o número de versión; y

por tipo de vulnerabilidad, gravedad, alcance de explotación relacionada e impacto (CVE, 2020).

Marco Teórico

Sistemas de Escaneo y Detección Temprana de Vulnerabilidades

Para conocer el estado del arte acerca de los sistemas de escaneo y detección temprana de vulnerabilidades se realiza una revisión sistemática de literatura basada en la guía metodológica de Barbara Kitchenham (Kitchenham & Charters, 2007).

Para el proceso de revision sistematica de literatura se plantea cinco preguntas:

- RQ1. ¿Cuáles son los enfoques de detección de vulnerabilidades?
- RQ2. ¿Cómo evaluar el rendimiento de las herramientas de detección de vulnerabilidades?
- RQ3. ¿Qué impacto tiene el tiempo en la detección de vulnerabilidades?
- RQ4. ¿Cuáles son los procesos de desarrollo para implementar un software de análisis de vulnerabilidades?
- RQ5. ¿Cuáles son los retos y las futuras direcciones de investigación en la detección de vulnerabilidades?

Criterios de inclusión y exclusión. Para la selección de estudios se tomó en cuenta aquellos que fueron publicados a partir del año 2015, debido a que existe una constante actualización de herramientas, procesos de desarrollo y aplicación a nivel de ciberseguridad. Además, se definió los siguientes criterios de inclusión:

- Artículos cuya revista de publicación esté en un cuartil Q3 o superior.

- Artículos que evalúen cuantitativamente el rendimiento de los sistemas, prototipos o herramientas.

Para los criterios de exclusión se considó los siguientes parámetros:

- Estudios relacionados con Healthcare
- Estudios que no presenten contenido en Ingles

Proceso de búsqueda. Para el proceso de búsqueda se utilizó las siguientes bases de datos de investigación académica: IEEE Xplore, SCOPUS, ScienceDirect y ACM. Se realizó una revisión bibliográfica para obtener un grupo de palabras que permitiera buscar estudios científicos relacionados con el tema de investigación. Para las búsquedas iniciales, se utilizó los siguientes términos:

- ”Early detection of vulnerabilities.”
- ”Vulnerability scanning systems.”
- ”Vulnerability scanning tools.”
- ”Alerts or vulnerability treatment.”
- ”Development processes for vulnerability analysis.”

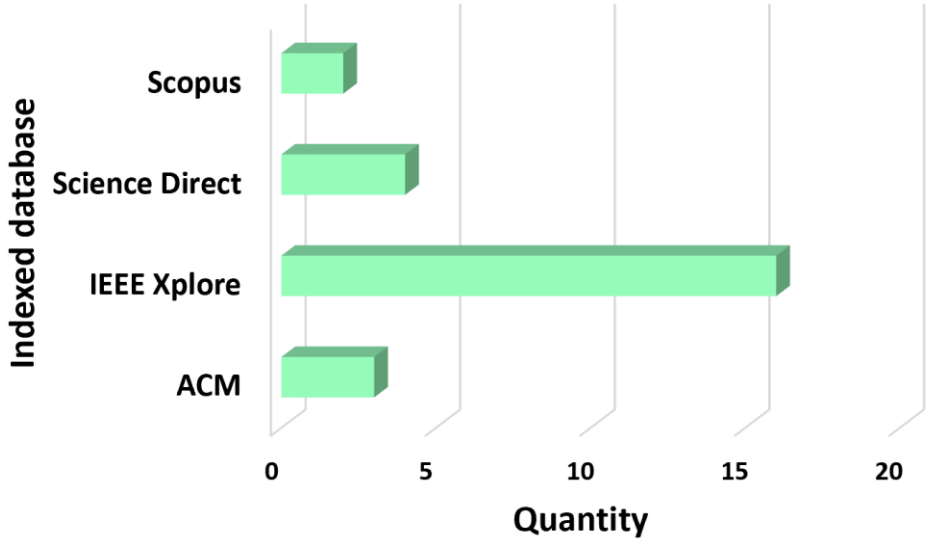
Posteriormente, con base en los términos mencionados, se afinó y se obtuvo la siguiente cadena de búsqueda: ((“Development Process”) And (”Vulnerability Detection” Or ”Vulnerability Scanning” Or ”Vulnerability Monitoring” Or ”Vulnerability Analysis”) And (”Tool” Or ”System”) And (”Data Science” Or ”Deep Learning” Or ”Data Mining” Or ”Artificial Intelligence”)). , con la cual se logró recopilar 135 artículos, los cuales fueron cargados a la herramienta Ryyan - Intelligent Systematic Review, una aplicación web de uso gratuito que permite automatizar el proceso de revisión sistemática.

Una vez aplicados los criterios de inclusión y exclusión se obtuvo 45 artículos. Finalmente, a través de un proceso de lectura se realizó una última revisión, mediante un proceso de lectura exhaustiva se obtuvo 25 artículos que fueron seleccionados como estudios primarios.

Análisis de los estudios primarios. Los 25 estudios primarios seleccionados pertenecen al área de la informática, se consideraron los artículos de revistas que cumplían los criterios de inclusión de la búsqueda y los que pertenecían a SCImago Journal Rank en un cuartil Q3 o superior. La Figura 1 muestra el número de artículos de cada base de datos de investigación académica.

Figura 1

Artículos publicados por base de datos indexada.



Nota. La figura representa el número de artículos en las principales bases de datos

Las revistas donde se publicaron los articulos seleccionados se distribuyen de la siguiente forma: IEEE Access (Zhang & Sakurai, 2021) (Liu & Wang, 2018) (Yi, Xu, & Xu, 2017) (Han, Zhou, Qian, Fu, & Zou, 2019) (Zagane, Abdi, & Alenezi, 2020) (Sõnmez & Kilic, 2021) (Zeng, Lin, Pan, Tai, & Zhang, 2020) (Qin, y otros, 2016) , IEEE Transactions on Information Forensics and Security (H., y otros, 2021), IEEE

Transactions On Services Computing (Antunes & Vieira, 2015), IEEE Transactions on Fuzzy Systems (Liu, y otros, 2019), ACM Computing Surveys (Qasemen, y otros, 2021) (Meng, Liu, Zhang, Pokluda, & Boutaba, 2015) (Ghaffarian & Shahriari, 2017), ELSEVIER Applied Intelligence (C.B., Ö.B., & Abualigah, 2021), ELSEVIER Computers & Security (Jeon & Kim, AutoVAS: An automated vulnerability analysis system with a deep learning approach, 2021), ELSEVIER Information and Software Technology (Cao, Sun, Bo, Wei, & Li, 2021), ELSEVIER The Journal of Systems & Software (Zheng, y otros, 2020), Future Internet MDPI (Yu, Zhuge, Cao, Shi, & Jiang, 2020).

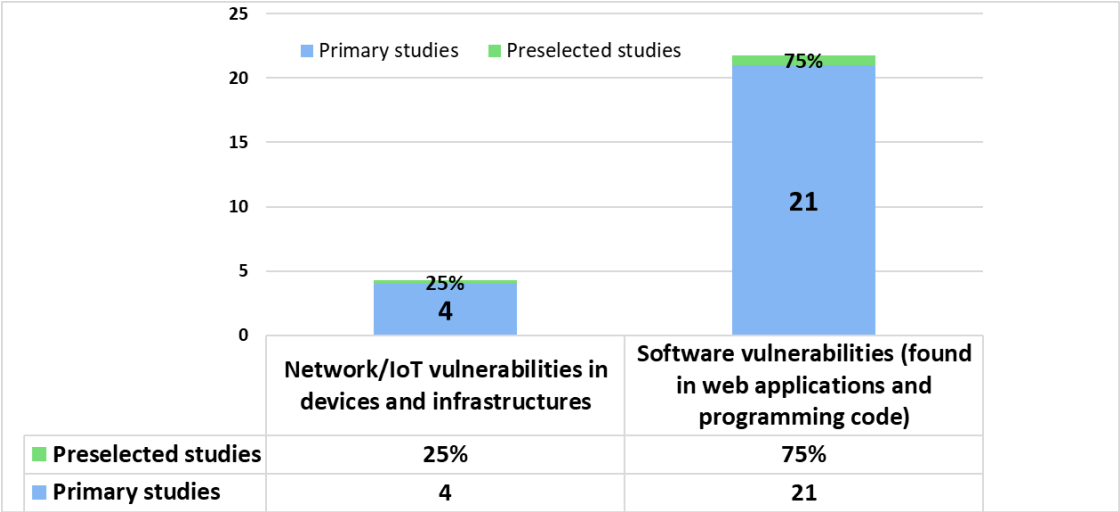
De acuerdo con la RQ1, los estudios demostraron dos enfoques de VDAs (Vulnerability Detection Approaches). En el análisis del estado del arte, los VDAs representan el entorno en el que se detectan las vulnerabilidades y el proceso concreto que se aplica. Los dos principales VDAs identificados son:

- Vulnerabilidades de software (encontradas en aplicaciones web y código de programación) (Antunes & Vieira, 2015) (Cao, Sun, Bo, Wei, & Li, 2021) (Cigoj, 2019) (Ghaffarian & Shahriari, 2017) (Han, Zhou, Qian, Fu, & Zou, 2019) (Jeon & Kim, 2021) (Li, 2019) (Liu, y otros, 2019) (Liu & Wang, 2018) (Qiang, 2017) (Qin, y otros, 2016) (Ren, 2019) (S¸ahin, 2021) (Sõnmez & Kilic, 2021) (Wang H. Y., 2021) (Yu, Zhuge, Cao, Shi, & Jiang, 2020) (Zagane, Abdi, & Alenezi, 2020) (Zeng, Lin, Pan, Tai, & Zhang, 2020) (Zhang & Sakurai, 2021) (Zheng, y otros, 2020) (Zhu, 2020).
- Vulnerabilidades de red/IoT en dispositivos e infraestructuras (Meng, Liu, Zhang, Pokluda, & Boutaba, 2015) (Qasemen, y otros, 2021) (Yi, Xu, & Xu, 2017) (Yu, Zhuge, Cao, Shi, & Jiang, 2020).

Durante el proceso de selección del estudio, se observó una tendencia creciente de investigar las vulnerabilidades del software en comparación con la otra categoría. La Figura 2 ilustra el porcentaje de estudios preseleccionados y el número de estudios primarios correspondientes a cada enfoque. Debido a que los VDAs determinan amplios campos de estudio y análisis, se clasificaron en función de su ID, que representa la referencia bibliográfica del estudio, las magnitudes de detección que muestran el tipo de vulnerabilidades que son analizadas dentro del enfoque de detección, las técnicas o herramientas de análisis que se utilizaron en el proceso de detección y, por ultimo, las principales características de los estudios. (Ver Tabla 1, Tabla 2, Tabla 3).

Figura 2

Distribución de los estudios preseleccionados y primarios a los VDAs



Nota. La figura muestra la tendencia de los estudios resultantes de acuerdo con el enfoque de detección.

Tabla 1

Vulnerabilidades de Software

Cita	Título	Magnitud de detección	Método de detección	Técnicas / Herramientas de análisis	Principales Características
(Zhang & Sakurai, 2021)	A Survey of Software Clone Detection From Security Perspective	Clones de código de software	Deep learning-based code clone vulnerability detection, vulnerable code clone detection for 5G-Internet of Things devices, real-time detection methods	Análisis dinámico Análisis estático	Este estudio analiza la detección de clones de código de software desde el punto de vista de seguridad, ya que los ciberdelincuentes se enfocan principalmente en la codificación de las aplicaciones web, por lo cual los autores mencionan que un análisis dinámico en tiempo real permite una mayor seguridad.
(Han, Zhou, Qian, Fu, & Zou, 2019)	An Optimized Static Propositional Function Model to Detect Software Vulnerability	Vulnerabilidades en código	Static detection model based on the proposition function	Control Flow Graph (CFG) and program dependency graph (PDG)	La detección automática de vulnerabilidades es una tendencia, los autores mencionan que el principal defecto en la detección estática es la falta de criterios en la definición de las propiedades de las vulnerabilidades, es decir características y reglas que las definen como tal y hacen que los sistemas tomen las decisiones acertadas
(Zagane, Abdi, & Alenezi, 2020)	Deep Learning for Software Vulnerabilities Detection Using Code Metrics	Vulnerabilidades en código	Deep Learning and Machine Learning, Vulnerability Prediction Model (VPM)	Standard k-folds cross-validation technique	Descubrir la ubicación exacta de código vulnerable reduce el tiempo de verificación, además este estudio muestra que las métricas de código permiten a Deep Neural Networks (DNN) aprender más características de código vulnerable.

Cita	Título	Magnitud de detección	Método de detección	Técnicas / Herramientas de análisis	Principales Características
(Zeng, Lin, Pan, Tai, & Zhang, 2020)	Software Vulnerability Analysis and Discovery Using Deep Learning Techniques: A Survey	Vulnerabilidades en código	Deep Learning	DNN (Deep Belief Network), BiLSTM, LSTM, Bidirectional RNN, Three concatenated CNNs	Este estudio realiza la revisión de estudios enfocados específicamente en Deep Learning para la detección de vulnerabilidades, su principal conclusión es que a través de los estudios existentes todavía no se ha alcanzado una madurez en este tema, ya que es un proceso constante.
(Qin, y otros, 2016)	Vulnerability Detection on Android Apps–Inspired by Case Study on Vulnerability Related with Web Functions	Vulnerabilidades causadas por el mal uso de las API	Heuristic Method	Vularcher-A Tool To Detect Vulnerabilities	En este estudio se analizaron cuatro tipos de vulnerabilidades: Improper certificate validation, WebView bypass certificate validation vulnerability, WebView remote code execution vulnerability, Alibaba Cloud OSS credential disclosure vulnerability, las cuales permiten el ataque de tipo Man in the middle y propone un método de identificación de vulnerabilidades a través de la herramienta VulArcher.
(H., y otros, 2021)	Combining Graph-Based Learning with Automated Data Collection for Code Vulnerability Detection	Vulnerabilidades en el código fuente del programa	FUNDED (Flow-sensitive vUl-Nerability coDE Detection)	Graph Neural Networks (GNNs)	En este estudio se combina el aprendizaje probabilístico y las evaluaciones estadísticas para recopilar muestras de proyectos de código abierto, las cuales permiten tener indicadores previos para evaluar vulnerabilidades en el código.

Cita	Título	Magnitud de detección	Método de detección	Técnicas / Herramientas de análisis	Principales Características
(Liu, y otros, 2019)	DeepBalance: Deep-Learning and Fuzzy Oversampling for Vulnerability Detection	Desequilibrio de clases en la detección de vulnerabilidades de software	Machine Learning, DeepBalance system, which combines the new ideas of deep code representation learning and fuzzy-based class rebalancing.	Deep neural network with Bidirectional Long Short-Term Memory (BiLSTM)	El principal problema al que se enfrenta Machine Learning es detectar si el código es vulnerable o no, deep learning and fuzzy oversampling permite lidiar con este problema de desequilibrio en el código.
(Ghaffarian & Shahriari, 2017)	Software Vulnerability Analysis and Discovery Using Machine-Learning and Data-Mining Techniques: A Survey	Vulnerabilidades en código	Machine-learning and data-mining	-Supervised Learning, -Unsupervised Learning -Reinforcement Learning -Vulnerability Prediction Models based on Software Metrics -Anomaly Detection Approaches -Vulnerable Code Pattern Recognition -Miscellaneous Approaches	Los autores mencionan que Machine-learning and data-mining es un campo con estudios limitados en el enfoque de detección de vulnerabilidades, sin embargo, los estudios analizados muestran resultados prometedores en el descubrimiento y análisis de vulnerabilidades de software.

Cita	Título	Magnitud de detección	Método de detección	Técnicas / Herramientas de análisis	Principales Características
(C.B., Ö.B., & Abualigah, 2021)	Prediction of software vulnerability based deep symbiotic genetic algorithms: Phenotyping of dominant-features	Vulnerabilidades en código	Metaheuristic Optimization Methods, Deep Learning Method and SYMbiotic Genetic algorithm	Deep Symbiotic-Based Genetic Algorithm Model (DNN-SYMbiotic GAs)	Los autores realizan un estudio avanzado el cual propone evaluar la predicción de vulnerabilidades y la calidad del software, los autores mencionan que no basta con detectar la vulnerabilidad, es fundamental predecirla en base a la codificación que realiza el desarrollador.
(Jeon & Kim, AutoVAS: An automated vulnerability analysis system with a deep learning approach, 2021)	AutoVAS: An automated vulnerability analysis system with a deep learning approach	Vulnerabilidades en código fuente	Deep Learning-Based Automated Vulnerability Analysis System (AutoVAS)	Automated vulnerability scanning based on source codes from National Vulnerability Database (NVD) and Software Assurance Reference Database (SARD) Deep Learning model was proposed from the viewpoints of program slicing and embedding techniques.	Deep Learning basado en hojas de datos de otros proyectos de detección de vulnerabilidades permite un proceso de identificación automatizado que a su vez brinda la posibilidad de tener un campo más amplio de detección, al llegar a encontrar vulnerabilidades zero-day.

Cita	Título	Magnitud de detección	Método de detección	Técnicas / Herramientas de análisis	Principales Características
(Cao, Sun, Bo, Wei, & Li, 2021)	BGNN4VD: Constructing Bidirectional Graph Neural-Network for Vulnerability Detection	Vulnerabilidades en código fuente	Deep Learning - BGNN4VD (Bidirectional Graph Neural Network for Vulnerability Detection)	Construcción de una red neuronal gráfica bidireccional (BGNN)	BGNN4VD logra tener una mayor precisión y exactitud cada vez que se aplican las ultimas vulnerabilidades reportadas por CVE, esto demuestra que el aprendizaje constante brinda un campo de evaluación mayor.
(Zheng, y otros, 2020)	The impact factors on the performance of machine learning-based vulnerability detection: A comparative study	Vulnerabilidades en código	Machine Learning-Based and Deep Learning-Based Vulnerability Detection Methods	-Traditional Machine Learning (Random Forest (RF) -Gradient Boosting Decision Tree (GBDT) -K-Nearest Neighbors (KNN) -Support Vector Machine (SVM) -Logistic Regression (LR)) -Deep learning (Bi-directional LSTM (BLSTM) -Gated Recurrent Unit (GRU) -Convolutional Neural Networks (CNN))	Los autores presentan cuatro factores de impacto que pueden afectar directamente al rendimiento de la detección de vulnerabilidades. Y mencionan que a medida que se proporcionan más datos relacionados con vulnerabilidades mejora el rendimiento de detección.

Nota. La tabla representa un resumen de los estudios analizados de acuerdo con vulnerabilidades de software

Tabla 2

Vulnerabilidades de IOT y Red

Cita	Título	Magnitud de detección	Método de detección	Técnicas / Herramientas de análisis	Principales Características
(Yi, Xu, & Xu, 2017)	An Intelligent Communication Warning Vulnerability Detection Algorithm Based on IoT Technology	Vulnerabilidades en entorno IOT	Static detection method for early warning vulnerabilities based on the counterexample of IoT	Vulnerability Detection Attack Pattern Algorithm for IoT Dynamic Model Based on Intelligent Communication Early Warning Vulnerability Detection Static detection of buffer warning vulnerabilities in the anti-example of the IoT	Para que exista una alerta temprana, debe existir un monitoreo constante de la red, a través de una detección estática basada en la asociación de ataques y estudio del algoritmo de ataque se puede predecir y evaluar la red basado en el nivel de seguridad. El número de vulnerabilidades fue comparado a través de una prueba que relaciona Intelligent Warning con Vulnerability Mining Detection.
(Qasemen, y otros, 2021)	Automatic Vulnerability Detection in Embedded Devices and Firmware: Survey and Layered Taxonomies	Vulnerabilidades en el código binario y el firmware de los sistemas embebidos desplegados	Taxonomías basadas en las aplicaciones de estos enfoques, las características y el tipo de análisis.	Análisis dinámico Análisis estático Ejecución simbólica Enfoques híbridos	Debido a que el firmware que controla varios de los dispositivos IOT se vuelven obsoletos en un tiempo limitado, es importante mantener diferentes tipos de análisis que permiten tener un mayor control de la infraestructura IOT, para lograr una seguridad controlada, esto a través de varias metodologías.

Cita	Título	Magnitud de detección	Método de detección	Técnicas / Herramientas de análisis	Principales Características
(Meng, Liu, Zhang, Pokluda, & Boutaba, 2015)	Collaborative Security: A Survey and Taxonomy	Seguridad colaborativa en entornos distribuidos	Collaborative Intrusion Detection Collaborative Anti-Spam, Collaborative Anti-Malware Collaborative Identification of Malicious Nodes Collaborative Malware Detection in Mobile OS Collaborative Detection and Resistance to Botnets	Analysis Target (Host Information, Network Traffic) Timeliness of Analysis (Offline Analysis, Online Analysis) Architecture (Centralized, Decentralized, Hierarchical, Hybrid Decentralized) Network Infrastructure (Wired Network, Wireless Network) Initiative (Active Collaboration, Passive Collaboration) Shared Information (Raw Data, Partially Processed Data, Processed Data) Interoperability (Standard Communication, Customized Communication)	El proceso de compartir información es fundamental para asegurar la precisión en la detección de ataques, aporta en la capacidad de los sistemas ante ataques sofisticados de los cuales no se tenía conocimiento, por lo cual la seguridad colaborativa es una tendencia.

Cita	Título	Magnitud de detección	Método de detección	Técnicas / Herramientas de análisis	Principales Características
(Yu, Zhuge, Cao, Shi, & Jiang, 2020)	A Survey of Security Vulnerability Analysis, Discovery, Detection, and Mitigation on IoT Devices	Vulnerabilidades en dispositivos IOT / Superficies de ataque	Network Scanning Method Similarity Detection Method	Análisis dinámico (Fuzzing, Taint checking) Análisis estático (symbolic execution, taint analysis, data-flow analysis)	Se realiza un análisis de la arquitectura IOT y las superficies de ataque desde el punto de vista del consumidor y de la industria, se determina que el núcleo de la seguridad es la vulnerabilidad, las cuales son investigadas principalmente a través de un escaneo de red y la detección de similitudes de código.

Nota. La tabla representa un resumen de los estudios analizados de acuerdo con vulnerabilidades de IOT y red

Tabla 3

Vulnerabilidades en aplicaciones y servicios web

Cita	Título	Magnitud de detección	Método de detección	Técnicas / Herramientas de análisis	Principales Características
(Liu & Wang, 2018)	A Web Second-Order Vulnerabilities Detection Method	Vulnerabilidades de seguridad de segundo orden de la web	Rastreo de URL	Black box second-order vulnerability detection algorithm	En este artículo se menciona que las vulnerabilidades de segundo orden son más destructivas que las de primer orden, por lo cual el algoritmo de detección automatizado que han diseñado cumple satisfactoriamente con el proceso de rastreo de vulnerabilidades en las URL.
(Sõnmez & Kilic, 2021)	Holistic Web Application Security Visualization for Multi-Project and Multi-Phase Dynamic Application Security Test Results	WEB APPS VULNERABILITIES	OWASP ZAP scanner tool	Metrics/Measures for the multi-project/phase environment	Este estudio proporciona una vista técnica y una vista de gestión, a través de los resultados de varios escáneres de vulnerabilidades de aplicaciones web, de los cuales se extrae un amplio conjunto de medidas/métricas que permiten tanto la evaluación como la gestión de vulnerabilidades. Además, se menciona que para los expertos en software es importante tener graficas que les permitan tomar decisiones basado en las métricas propuestas.
(Antunes & Vieira, 2015)	Assessing and Comparing Vulnerability Detection Tools for Web Services: Benchmarking Approach and Examples	Entornos de servicios web	Penetration testing Analizadores de código estático Detectores de anomalías	SQL Injection	Los autores de este estudio consideran que el proceso de selección de herramientas para el análisis y detección de vulnerabilidades debe ser un proceso técnico donde la comparativa se base en métricas que logren definir la eficiencia y capacidad de detección ya que los enfoques de detección que tienen las herramientas comerciales suelen ser limitados y dejan varios dominios sin evaluar.

Nota. La tabla representa un resumen de los estudios analizados de acuerdo con vulnerabilidades en aplicaciones y servicios web

La clasificación de la información bajo estos parámetros permitió identificar que el 96% de los estudios definen fases de detección de vulnerabilidades. En los estudios que han definido fases de detección, se implementa un prototipo (48%), una herramienta (29%) o un sistema (23%) se implementa para las pruebas funcionales. No hay una norma en las fases aplicadas, pero se identifican fases representativas independientemente del VDA o del método.No se identificaron metodologías para la detección de vulnerabilidades, pero según las fases y características principales, las metodologías deben ser iterativas para lograr un seguimiento continuo.

En cuanto a la RQ2, el rendimiento de las herramientas de detección de vulnerabilidades se evalúa con estadísticas ampliamente utilizadas de análisis predictivo (Antunes & Vieira, 2015) (Cao, Sun, Bo, Wei, & Li, 2021) (Ghaffarian & Shahriari, 2017) (Jeon & Kim, 2021) (Liu, y otros, 2019) (Zagane, Abdi, & Alenezi, 2020) (Zhang & Sakurai, 2021) (Antunes & Vieira, 2015) (Zheng, y otros, 2020).

Las métricas de evaluación empleadas son las siguientes:

- Verdaderos positivos (TP) que es el número de vulnerabilidades verdaderas detectadas.
- True Negative (TN) que indica el número de muestras que no son vulnerables y se detectan como no vulnerables.
- Falsos Positivos (FP) que expresa el número de vulnerabilidades vulnerabilidades detectadas que no existen.
- Falsos Negativos (FN) que indica el número de muestras vulnerables que se detectan como no vulnerables.
- Exactitud (A) que determina la veracidad de todo lo detectado.
- Precisión (P) que determina la exactitud de las vulnerabilidades detectadas.

- Recall (R) que determina el número de muestras verdaderas positivos en el total de muestras vulnerables.
- Puntuación F1 o medida F1 (F1) (Vielberth, Bohm, Fichtinger, & Günther, 2020), (IBM Corporation, 2020) que determina la eficiencia global al considerar la precisión como la tasa de FN.
- La Especificidad o Tasa de Falsos Negativos (FNR) se centra en el algoritmo o modelo y evalúa los casos negativos que ha clasificado correctamente, midiendo la eficacia de la detección; la tasa de falsos positivos (FPR) mide el porcentaje de falsos positivos frente a todas las predicciones positivas (IBM Corporation, 2020) (Zheng, y otros, 2020).

De acuerdo con la RQ3, el tiempo es el factor fundamental en el cual se centran cada uno de los estudios, la capacidad de detección y respuesta a incidentes de seguridad informática implica una variedad de decisiones e innovaciones las cuales permitan organizar y sistematizar la información (Yi, Xu, & Xu, 2017). Las vulnerabilidades no tratadas pueden convertirse en incidentes de seguridad informática, que tienen efectos fatales en las organizaciones.

Entre los incidentes más comunes están la mala configuración de las páginas web, los servicios fuera de línea, la exposición de información sensible, la infección de servidores, el acceso no autorizado y el secuestro de datos. Estos incidentes pueden desestabilizar y desacreditar las infraestructuras tecnológicas. Las vulnerabilidades tempranas es vital para evitar que la ciberdelincuencia se haga realidad (Liu & Wang, 2018).

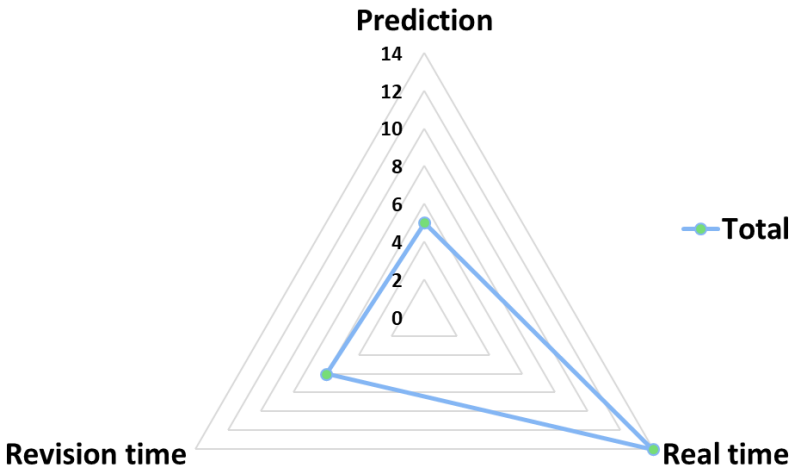
Algunos estudios presentan propuestas ambiciosas que intentan predecir vulnerabilidades (C.B., Ö.B., & Abualigah, 2021) o encontrar vulnerabilidades zero-day (Jeon & Kim, 2021). Otros estudios llevan a cabo procesos de revisión mediante auditorías que comparan los resultados con herramientas comerciales. Sin embargo,

en la mayoría de estudios se intenta encontrar las vulnerabilidades en tiempo real (Zhang & Sakurai, 2021), (Cao, Sun, Bo, Wei, & Li, 2021).

La Figura 3 representa la tendencia de los principales estudios en relación con los tres tiempos de detección. El principal objetivo del VDS es detectar vulnerabilidades en el mínimo tiempo posible. Por otro lado, estos fallos de software o hardware deben ser mitigados, para evitar que sean explotados con fines maliciosos que afecten a las organizaciones.

Figura 3

Tendencia del tiempo en la detección de vulnerabilidades



Nota. La figura representa los tres tiempos de deteción en los estudiios analizados.

En relación con la RQ4, no se ha identificado ningún proceso de desarrollo estándar. Según los estudios revisados, fue posible extraer un proceso general basado en fases de detección, a pesar de los diferentes enfoques y métodos empleados que se muestran en la Tabla 1, 2 y 3. El proceso comienza definiendo si el objetivo es detectar vulnerabilidades específicas, en cuyo caso es necesario entender primero los principios del ataque examinado (Liu & Wang, 2018).

Por otro lado, se propone el uso eficiente de herramientas de detección (Antunes & Vieira, 2015), nuevos métodos o algoritmos (Liu, y otros, 2019) (Wang H. Y., 2021). Una vez que el enfoque de detección está claro, se diseña la arquitectura del sistema (Cao, Sun, Bo, Wei, & Li, 2021) (Jeon & Kim, 2021) (Liu & Wang, 2018) (Ren, 2019) (Yi, Xu, & Xu, 2017) (Yu, Zhuge, Cao, Shi, & Jiang, 2020). Para implementar el VDS, se debe aplicar una metodología de desarrollo evolutiva o incremental.

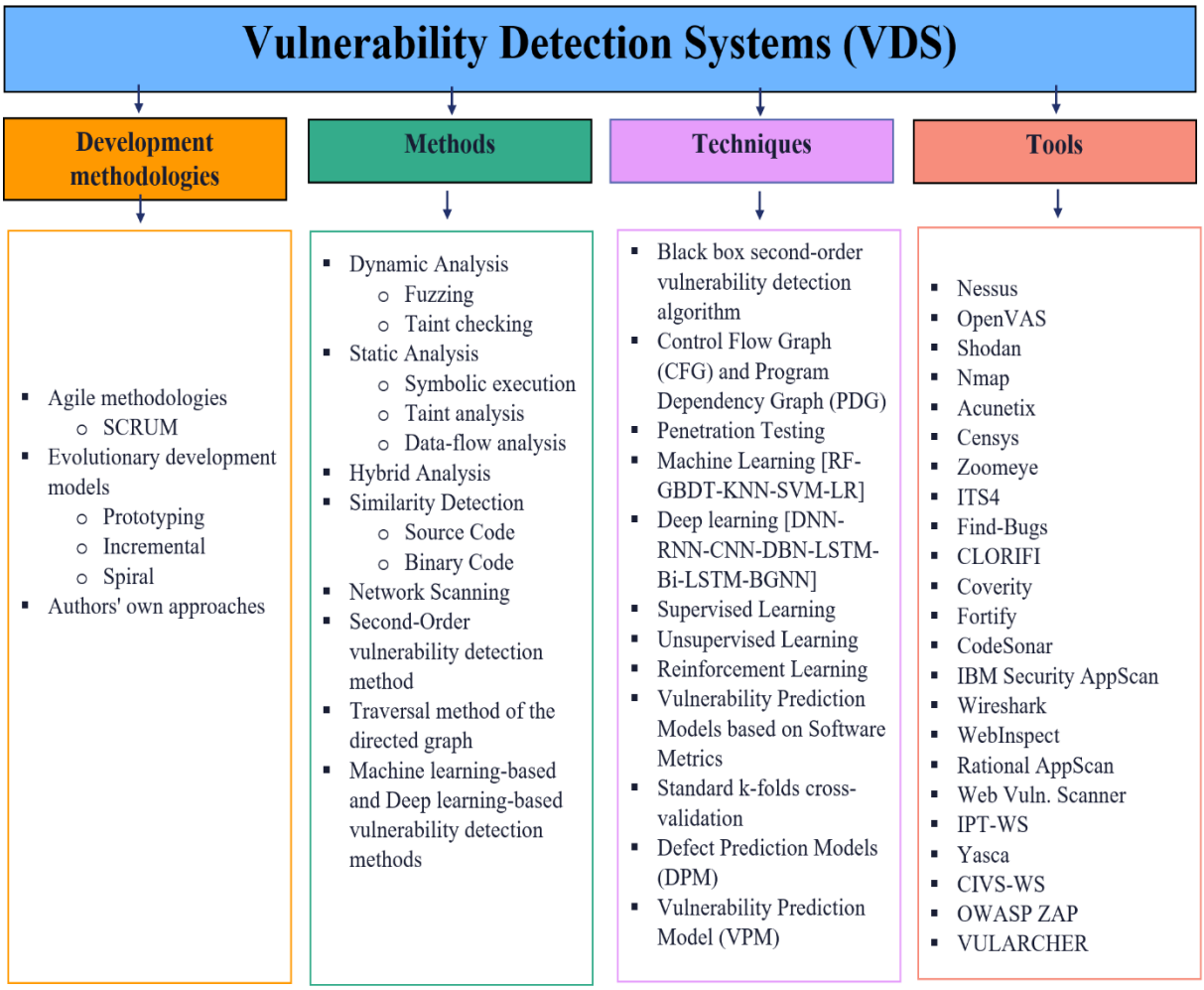
La mejor metodología para el desarrollo de este tipo de sistemas aún no ha sido definida. Sin embargo, hay que tener en cuenta que el VDS tiene que ser validado y está sujeto a cambios constantes. Para validar el funcionamiento del VDS, se proponen inicialmente prototipos o herramientas antes de su implantación (Cigoj, 2019) (Qiang, 2017) (Ren, 2019) (Yu L. L., 2021) (Zhu, 2020). La evaluación del VDS se realiza mediante una comparación con los resultados de las herramientas comerciales (Antunes & Vieira, 2015) (Cao, Sun, Bo, Wei, & Li, 2021) (Qin, y otros, 2016) (Sõnmez & Kilic, 2021), al utilizar las métricas de evaluación presentadas en RQ2 (Antunes & Vieira, 2015) (Cao, Sun, Bo, Wei, & Li, 2021) (Ghaffarian & Shahriari, 2017) (Jeon & Kim, 2021) (Liu, y otros, 2019), (Zagane, Abdi, & Alenezi, 2020) (Zhang & Sakurai, 2021) (Zheng, y otros, 2020) (Zhu, 2020) o al combinar ambos métodos (Li, 2019) (Yu L. L., 2021) (Zhu, 2020). Por último, se definen las principales conclusiones del proceso de desarrollo y se proponen los cambios para afinar el VDS.

El proceso de desarrollo de software para la detección de vulnerabilidades requiere metodologías iterativas. Sin embargo, no existe una metodología de desarrollo claramente definida o un estándar. El proceso de desarrollo aplicado debe permitir cambios constantes para evaluar el rendimiento de los métodos y técnicas en el proceso de detección de vulnerabilidades. Además, se utilizan herramientas adecuadas para comparar la detección resultados o proporcionar los datos de entrada

para el VDS, que procesa y almacena la información relacionada con las vulnerabilidades. La Figura 4 muestra información relacionada con las metodologías de desarrollo, métodos, técnicas y herramientas para el VDS.

Figura 4

Principales parámetros para el desarrollo de los VDS



Nota. La figura contiene los valores representativos de cada uno de los estudios

En este contexto, Liu y Wang (Liu & Wang, 2018) propusieron una arquitectura B/S que permite a los usuarios realizar eficientemente la detección de vulnerabilidades de segundo orden en aplicaciones aplicaciones web a través de los navegadores. B/S tiene un diseño modular que facilita la futura expansión del sistema. En su sistema de

alerta temprana de vulnerabilidades para el entorno del IoT (Yi, Xu, & Xu, 2017), Yi cargan datos de entrada de vulnerabilidades conocidas para hacer una asociación de ataques y detectar anomalías en la red. Jeon y Kim (Jeon & Kim, 2021) desarrollan un sistema llamado AutoVAS basado en el aprendizaje profundo que tiene a CVE y CWE como datos de entrada para prevenir amenazas en el software. Lu et al. (Yu, Zhuge, Cao, Shi, & Jiang, 2020) y Ren et al. (Ren, 2019) propusieron prototipos para verificar la eficacia de los métodos de detección y las arquitecturas de almacenamiento de vulnerabilidades a través de estudios de casos y comparaciones. mediante estudios de casos y comparaciones con otras herramientas comerciales. Como conclusión general, todo VDS busca optimizar los recursos y lograr un rendimiento que cumpla el objetivo de detección.

En cuanto a la RQ5, los principales retos en el desarrollo de estos sistemas son la complejidad de la detección y la definición de los parámetros e indicadores que determinan los Falsos Positivos y los Falsos Negativos (Han, Zhou, Qian, Fu, & Zou, 2019). Los métodos de Deep Learning, Machine Learning, Machine Learning y Data Mining agrupan grandes cantidades de datos y requieren considerables recursos de hardware (Liu, y otros, 2019). El desarrollo de VDS y los procesos de tratamiento de vulnerabilidades de vulnerabilidad representan un alto coste para las organizaciones. Al mismo tiempo, el rápido progreso tecnológico trae consigo nuevas vulnerabilidades; por lo tanto, los sistemas deben ser escalables y tener una mayor capacidad de integración con nuevos métodos y técnicas de detección (Meng, Liu, Zhang, Pokluda, & Boutaba, 2015).

En cuanto a las futuras direcciones de investigación, existe un amplio campo por investigar. Dado que no existe un estándar para desarrollar VDS, se pueden investigar y proponer procesos, metodologías, técnicas y herramientas relevantes. La inteligencia artificial (IA) requiere una consideración detallada en los procesos de

detección. Los nuevos VDS deben basarse en la seguridad colaborativa. La obtención de varios vectores para los datos de entrada permite un mayor campo de conocimiento y evaluación sobre las vulnerabilidades (Meng, Liu, Zhang, Pokluda, & Boutaba, 2015) (Jeon & Kim, 2021) (Cao, Sun, Bo, Wei, & Li, 2021).

Las técnicas de Deep learning mencionadas en los sistemas que detectan vulnerabilidades a nivel de software pueden ser afinadas mediante la definición de indicadores que puedan determinar con mayor precisión las FP y FN (Liu, y otros, 2019). Por último, existen potentes herramientas (por ejemplo, Nessus, Shodan, Nmap, Wireshark, Libav, Retina, etc.) que pueden facilitar el tratamiento de las vulnerabilidades encontradas (Yu, Zhuge, Cao, Shi, & Jiang, 2020), a través de nuevos sistemas sistemas distribuidos que consumen los servicios ofrecidos (APIs).

Priorización de vulnerabilidades

El estándar mundial usado por organizaciones en todo el mundo es CVSS, que se ha mantenido en constante evolución e investigación. CVSS consta de tres grupos de metricas: Base, Temporal y Ambiental (CVSS, 2019). Sin embargo, su constante desarrollo e investigacion es muestra de la dificultad que representa estandarizar el riesgo e impacto, y sobre todo que este aplique de igual manera para todas las infraestructuras tecnológicas, para lograr una priorización acertada, la cual signifique una distribución de recursos optima. Los primeros trabajos intentaban relacionar las carácterísticas propias de la vulnerabilidad, para determinar la probabilidad de explotación (Allodi & Massacci, Comparing Vulnerability Severity and Exploits Using Case Control Studies, 2014) .

Dondo (Dondo M. G., 2008) presenta un enfoque de sistemas difusos basados en los atributos de la vulnerabilidad que ayudan a evaluar el riesgo potencial relativo de los activos de la red informática. Usa como datos de entrada los CVE mediante las

bases de datos de vulnerabilidades conocidas como NVD. Utiliza las reglas difusas para hacer una inferencia sobre la exposicion al riesgo y la probabilidad de ataque, esto permite clasificar las vulnerabilidades y definir la prioridad de atención. Menciona además que el enfoque de priorización tiene valores de gravedad mas significativos que los calculados por el CVSS. Uno de los objetivos fundamentales de los modelos de priorizacion es lograr procesos automatizados para obtener resultados tengan altos porcentajes de efectividad.

Amankwah propone un nuevo marco automatizado para evaluar la gravedad de las vulnerabilidades de los escaneres web de código abierto. En este estudio menciona que CVSS es criticado por la alta sensibilidad, pero baja especificidad para los exploits utilizados y por ende la inconsistencia en la puntuacion de gravedad (R. Amankwah, 2020). Por esta razón, proponen cuatro referencias de evaluación: impacto, exploitabilidad, prevalencia and detectabilidad

El framework de este estudio tiene tres componentes: detección de vulnerabilidades, evaluación de vulnerabilidades basado en las referencias mencionadas y finalmente determina la severidad para priorizar las vulnerabilidades. Los resultados muestran un enfoque mas centralizado y una mayor precisión de riesgo que permite priorizar la atencion de vulnerabilidades.

Sharma and Singh por su parte proponen un enfoque híbrido derivado de la combinación de Vulnerability Rating and Scoring System (VRSS) y CVSS con dos metricas temporales: remediation level and vulnerability index (Sharma & Singh, 2018). En ese estudio se obtiene puntuación cuantitativa a partir de variables cualitativas para la priorizacion. Los datos de entrada son recopilados de NVD. Toman en consideración el nivel de remediación y la tasa a la que un tipo particular de vulnerabilidad esta creciendo con el tiempo, denominada índice de vulnerabilidad.

El índice de vulnerabilidad, eleva la puntuación estática para tener en cuenta la tasa de cambio de la vulnerabilidad con el tiempo. Spanos propone un proceso de text mining, que analiza la descripción que emite NVD respecto de las vulnerabilidades conocidas, donde aplican 3 métodos de clasificación: decision trees, neural networks and support vector machines, donde se concluye que la propia descripción es una fuente de información fiable y muy precisa para la priorización de la vulnerabilidad, en ese estudio lo que se intenta es definir si la descripción tiene relación con el CVSS que se define para cada vulnerabilidad (G. Spanos, 2017).

Sharma también realiza un proceso similar a través de convolution neutral network (CNN) donde intentan priorizar las vulnerabilidades basandose en palabras clave que se emiten en la descripcion, esto con el objetivo de definir un valor ya que inicialmente cuando una vulnerabilidad es presentada generalmente carece de un valor (R. Sharma, 2019). Deb and Roy presenta una implementacion matemática a través de una red bayesiana en el entorno software-defined networks para identificar el estado de los diferentes hosts en la red. El CVSS y la red bayesiana es una metodología solida para identificar la relación mutua entre las vulnerabilidades y para priorizar el proceso efectivo de atencion (Deb & Roy, 2020).

Por otro lado, Hu proponen dos algoritmos, el algoritmo de prediccion de amenazas basado en un gráfico bayesiano dinámico y el algoritmo de cuantificación de riesgos de seguridad basado en la prediccion de amenazas. El primer algoritmo tiene como objetivo proporcionar informacion de predicción completa con el escenario de la amenaza. El segundo algoritmo cuantifica la amenaza en el primer algoritmo en el riesgo de seguridad desde dos niveles: host y red. El framework que se propone en su estudio para cuantificar contiene 3 componentes: situational factors collection, threat scenario prediction and security risk situation quantification (Hu, Zhang, & Yang, 2018).

Chen mencionan que el tiempo promedio que tarda NIST en emitir un resultado sobre el riesgo de las vulnerabilidades descubiertas es alrededor de 132.7 días, por lo cual proponen un sistema llamado Vulnerability Analysis and Scoring Engine (VASE), el cual se basa en los resultados emitidos en los foros de ciberseguridad en twitter para recolectar las posibles puntuaciones y promediarlas, para obtener resultados tentativos acerca de la calificación para la vulnerabilidad en cuestión. VASE adopta el modelo graph convolutional network (GCN) en donde los nodos corresponden a CVEs. VASE se soporta en 3 componente principales: graph construction based on basic natural language processing methods, attention-based input embedding, and transductive inference with GCN-AE (Haipeng, Liu, Liu, Park, & Subrahmanian, 2019).

Es importante mencionar que, para varios de los estudios descritos anteriormente, los datos de entrada son masivos, es decir, los modelos aplicados se enfocan en todas las vulnerabilidades y en características propias de las mismas. Lo que se logra es aportar valores adicionales de evaluacion al modelo CVSS, que lo hagan mas específico al momento de definir el riesgo e impacto. Por otro lado, carecen de un enfoque que atienda variables de la organizacion en donde se puedan presentar las vulnerabilidades manteniendo la problematica de que un modelo tan genérico no atiende la especificidad necesaria para una priorizacion acertada (R. Amankwah, 2020), (A. Dobrovoljc, 2017), (Keramati, 2016).

Con el enfoque de dar solución al problema mencionado anteriormente, Farris presentan un software llamado VULCON el cual tiene una estrategia que se basa en dos metricas de rendimiento fundamentales: time to-vulnerability remediation (TVR) y total vulnerability exposure (TVE). Utiliza un mixed-integer multiobjective optimization algorithm para priorizar las vulnerabilidades. En su estudio analizan las variables de entorno donde se encuentran y detectan las vulnerabilidades. Las distintas variables

contenidas en las vulnerabilidades como el ano que se muestra en el CVE-ID son importantes al momento de priorizar, ademas de los puertos que se han configurado en las IPs de la red, ya que existe mayor probabilidad de aprovechamiento y explotacion (K. A. Farris, 2018).

VULCON intenta ir añadiendo variables de entorno que permitan evolucionar y afinar cada vez la calidad priorizacion. Mencionan que existe una mayor calidad en la priorización con las variables adicionales de entorno. Una de los factores fundamentales al momento de realizar una priorización acertada es la calidad de la información. Entre más específica sea esta, mayor es la probabilidad de lograr una priorizacion exitosa. Dado que se busca proponer modelos genericos la idea es basarse en la información que los mismos datos proporcionen (Sharma , Sibal, & Sabharwal, 2020), (Sharma & Singh, 2018).

Este enfoque hace que la priorizacion no sea sesgada en direccion alguna, atendiendo a un panorama de justificación al cual se acostumbran los expertos de ciberseguridad en las organizaciones para no atender aquellas vulnerabilidades que representan un “riesgo mínimo o no definido” o dejarlas en la línea del tiempo sin atender. Finalmente, los datos identificados en los procesos de escaneo seran los únicos predictores del riesgo real al que esta expuesta una organizacion.

Capítulo III

Marco Metodológico

Metodología y Proceso de desarrollo para los VDS

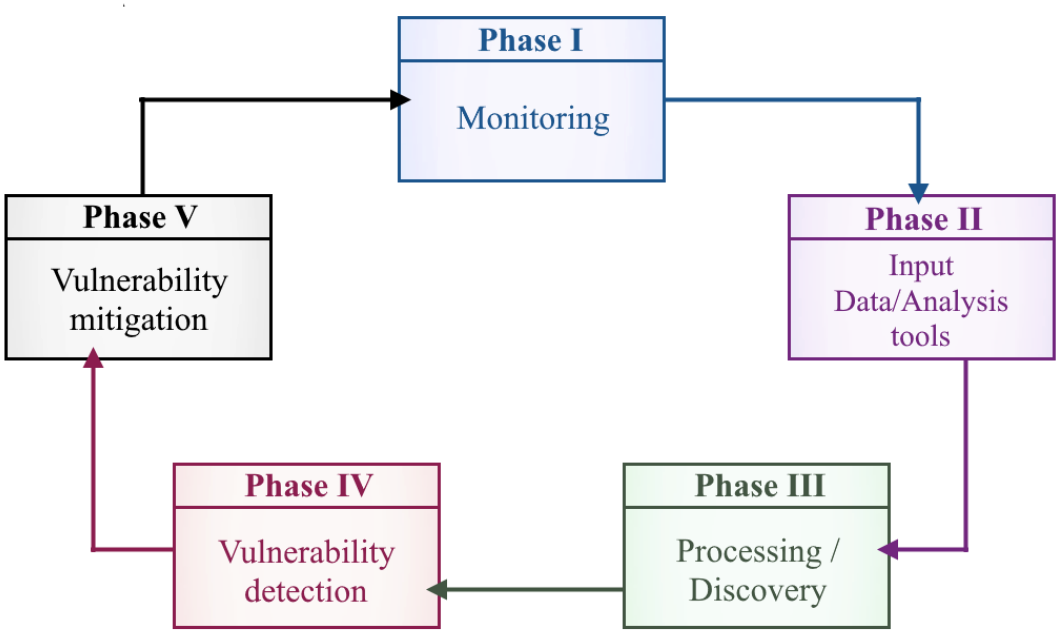
Basándose en los resultados del SLR, se divide la propuesta en dos importantes partes. La primera está relacionada con una metodología para la detección de vulnerabilidades. La segunda está relacionada con las fases del proceso de desarrollo, que cubren aspectos generales que se aplican a todos los sistemas de detección de vulnerabilidades. Los métodos, técnicas y herramientas para cada fase dependerán de las condiciones del entorno donde se aplicará el VDS.

Propuesta de metodología de detección de vulnerabilidades

Se propone una metodología de detección que consta de cinco fases generales fases generales que permiten detectar, analizar y mitigar las vulnerabilidades (ver Figura 5). Con la fase I, es posible obtener registros e información relacionada con una serie de eventos diferentes. En la fase II, se introducen los datos sobre las vulnerabilidades conocidas se introducen y los eventos se comparan con las vulnerabilidades conocidas públicamente mediante el uso de herramientas de análisis. En la fase III, se procesan los eventos conocidos y se descubren nuevos eventos se descubren a partir de la información obtenida anteriormente. En la fase IV, se definen y notifican los sucesos que se consideran vulnerabilidades. En la fase V intenta mitigar las vulnerabilidades mediante la corrección de los errores existentes. Todo el proceso se repite de forma iterativa para mantener una vigilancia constante

Figura 5

Fases de la metodología de detección de vulnerabilidades



Nota. La figura representa las cinco fases generales que contiene un proceso de deteccion y mitigacion de vulnerabilidades.

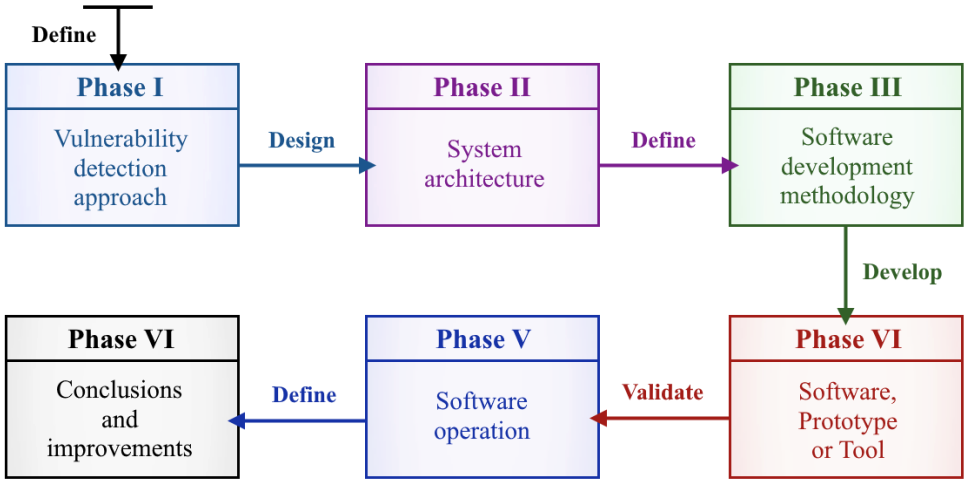
Propuesta de proceso de desarrollo del VDS

En cuanto al desarrollo, se propone un proceso que consta de seis fases (ver Figura 6). En la Fase I, se define el tipo de vulnerabilidades a detectar (es decir, el enfoque del VDS). En la Fase II, se determina los métodos, técnicas herramientas y recursos que se utilizarán en el desarrollo del VDS, con el objetivo de diseñar una arquitectura adaptada al enfoque de detección. En la fase III se fija una metodología de desarrollo y adapta o perfecciona el prototipo, la herramienta sistema. En la Fase IV, se configuran las herramientas, implementan las soluciones de software y se validan los errores. En la fase V, se aplica las métricas de evaluación o comparación de los resultados con herramientas comerciales con el mismo enfoque de detección, para validar el funcionamiento del software. Por último, en la Fase VI, se documenta y expone las principales conclusiones del proceso de desarrollo para establecer

correcciones que permitan un funcionamiento más eficiente del VDS. Es esencial evaluar constantemente el software, controlar la complejidad y los riesgos, y aumentar su funcionalidad a lo largo del proceso.

Figura 6

Fases del proceso de desarrollo de VDSs



Nota. La figura representa las fases principales que debe tener un proceso de desarrollo para los VDS.

Metodología multifactorial de recolección, análisis y toma de decisiones

La presente investigación aplica la metodología OSINT (Open Source Intelligence) para el proceso de coleccion, análisis y extracción de conocimiento de datos. A través del motor de busqueda Shodan se obtiene la información relevante acerca de vulnerabilidades existentes de diferentes IPs de un mismo dominio u organización. Debido a la gran cantidad de datos que puede devolver una consulta a traves de las REST API de Shodan, se aplicó ciertos algoritmos que permiten extraer la información relacionada solamente con vulnerabilidades.

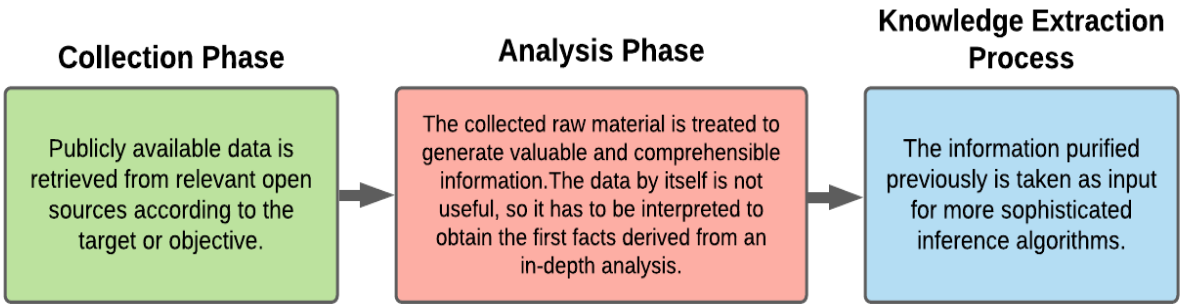
Luego con la información relevante se aplicaron ciertas fórmulas matemáticas para cuantificar las variables de entorno donde se descubrieron dichas vulnerabilidades. Una vez aplicadas las variables de entorno a la propia vulnerabilidad se puede procesar y priorizar el orden de atención. El objetivo de este proceso de busqueda es priorizar las vulnerabilidades que estan expuestas a cualquier usuario en internet y que se puede recopilar de forma abierta, de este modo se anade un valor agregado a la gestión de vulnerabilidades, al tomar en cuenta que el accionar de los ciberdelincuentes comienza por aquellos datos e informacion que se recopila abiertamente.

OSINT (Open Source Intelligence)

OSINT es una metodología que permite recopilar, procesar y correlacionar informacion de fuentes abiertas de todo el ciberespacio para generar conocimiento. Los avances tecnológicos hacen que OSINT evolucione a un ritmo vertiginoso, al proponer aplicaciones innovadoras impulsadas por datos y basadas en inteligencia artificial para diferentes áreas como: la política; la economía o la sociedad; pero tambien ofreciendo nuevas líneas de acción contra las ciber amenazas y el ciberdelito (Galindo, Nespoli, Gomez, & Martinez, 2020). OSINT tiene tres fases representativas que definen la metodología de procesamiento de la informacion, en la Figura 7 se describen sus fases

Figura 7

Fases OSINT



Nota. La figura representa uns descripción de las fases representativas en la metoodologia OSINT.

Fase de Colección

Para la fase de colección se utilizó Shodan, que es un escaner de seguridad informatica basado en la nube, posee un software cerrado, varios algoritmos de busqueda y se apoya de varias herramientas Open Source que proveen una amplia informacion sobre los hosts detectados por sus crawlers (Zolotykh, 2021). Se considera así mismo como el primer motor de busqueda del mundo para dispositivos conectados a internet (Shodan, 2021). En Shodan basta con ingresar el codigo org:”Organization Name” y se empieza a recopilar una serie de direcciones IP, a las cuales se las procesa y se obtiene información.

Shodan

Esta herramienta provee de información valiosa a los investigadores de ciberseguridad de las organizaciones, pero tambien a usuarios con intenciones maliciosas, ya que al ser un motor de busqueda está disponible a la comunidad en general (Lee, Shin, & Roh, 2017). Debido a la informacion que se puede recopilar de Shodan muchos expertos optan por usar tecnicas que oculten las diferentes IPs de los crawlers, de esta forma evitan ser detectados. Por otro lado, estan expertos que

hacen uso de esta valiosa informacion para aplicar medidas correctivas que mejoren la seguridad de sus redes. Shodan provee REST API para realizar consultas generales o específicas de acuerdo a las necesidades de los desarrolladores. Cuenta con una base de datos que almacena toda la informacion recopilada por sus crawlers acerca de las diferentes IP que rastrean en el gran mundo del internet. Shodan provee una respuesta en formato JSON lo cual facilita la manipulacion y extracción de datos.

Ademas, cuenta con un proceso de notificación automatizada, lo cual alerta y notifica los diferentes resultados que monitorea, cumpliendo con un servicio de deteccion temprana. Dependiendo de la IP escaneada y lo que en ella se detecte, Shodan puede mostrar alrededor de 11 variables generales a traves de un banner, que es el principal tipo de informacion que Shodan proporciona a través de la API REST. Dichas variables contienen a su vez otras propiedades con más 50 variables en total. En la Tabla 4 se detallan las 11 variables generales que son relevantes para este proceso de analisis. Esta información está directamente relacionada con el conocimiento y entendimiento de vulnerabilidades y del entorno donde se han identificado.

Tabla 4

Variables de especificación del banner de Shodan.

Variable	Descripción
Domains	Una matriz de cadenas que contiene los dominios de nivel superior para los nombres de host del dispositivo.
Hostnames	Una matriz de cadenas que contiene todos los nombres de host que se han asignado a la dirección IP de este dispositivo.
Org	El nombre de la organización que tiene asignado el espacio IP para este dispositivo.
Data	Contiene la información del banner del servicio.
City	El nombre de la ciudad donde se encuentra el dispositivo.

Variable	Descripción
Isp	El ISP que proporciona a la organización el espacio IP para este dispositivo. Considere esto como el "padre" de la organización en términos de propiedad de IP.
last_update	Fecha y hora de la última actualización/revisión de la IP
Vulns	Una matriz de cadenas que contiene el código CVE de las vulnerabilidades IP detectadas.
country_name	El nombre del país donde se encuentra el dispositivo.
ip_str	La dirección IP del host como una cadena.
Ports	El número de puerto en el que opera el servicio.

Nota. La tabla describe las variables que Shodan devuelve a través de su Rest API, en una consulta por IP

Durante esta fase se intenta obtener el detalle de cada una de las vulnerabilidades que han sido identificadas. A continuación se describe las variables que contiene una vulnerabilidad.

CVE (Common Vulnerabilities and Exposures). Es una lista de registros lanzada por MITRE Corporation en 1999, cada registro tiene un número de identificación, una descripción y al menos una referencia pública para las vulnerabilidades conocidas publicamente. Los registros CVE se utilizan en numerosos productos y servicios de ciberseguridad de todo el mundo. La sintaxis de CVE ID esta conformada por CV E pref ix + year + sequence number digits. Es importante conocer que la parte del año no indica cuando se descubrió la vulnerabilidad, sino solo cuando se hizo pública o se asigno. Mientras que la secuencia numérica son los identificadores únicos por año. Adicionalmente, CVE viene acompañado de una descripción única, la que generalmente incluye detalles como el nombre del producto y el proveedor afectados, un resumen de las versiones afectadas, el tipo de vulnerabilidad, el impacto, el acceso que requiere un atacante para explotar la vulnerabilidad y los componentes de codigo o entradas importantes que estan involucrados (CVE, 2021).

A continuación, se presenta un ejemplo de CVE:

- CVE ID: CVE-2017-9798
- CVE Description: Apache httpd allows remote attackers to read secret data from process memory if the Limit directive can be set in a user's .htaccess file, or if httpd.conf has certain misconfigurations, aka Optionsbleed. This affects the Apache HTTP Server through 2.2.34 and 2.4.x through 2.4.27. The attacker sends an unauthenticated OPTIONS HTTP request when attempting to read secret data. This is a use-after-free issue and thus secret data is not always sent, and the specific data depends on many factors including configuration. Exploitation with .htaccess can be blocked with a patch to the ap limit section function in server/core.c

Common Vulnerability Scoring System (CVSS). A cada CVE se le asigna una puntuacion que se trata de un sistema de puntaje diseñado para proveer un método abierto y estándar que permite estimar el impacto derivado de vulnerabilidades, contribuye a cuantificar la severidad que pueden representar las vulnerabilidades. Con este sistema de puntaje se obtienen valores de vulnerabilidad estandarizados para los CVE que van de 0 a 10, siendo 0 lo mas bajo y 10 considerado crítico, lo que permite mantener criterios consistentes para la gestión de las debilidades en hardware y software que han sido evaluadas. La puntuacion CVSS se calcula al combinar varias características de vulnerabilidad que se denominan métricas CVSS (CVSS, 2015). Al utilizar un marco abierto es posible conocer las características propias de cada vulnerabilidad. Sin embargo, al ser un panorama general no aborda variables propias de cada entorno donde se han identificado. Un método de esta naturaleza contribuye a tener un panorama amplio sobre la exposición

de una organizacion a riesgos, que pueden derivarse de las vulnerabilidades que ya han sido identificadas y evaluadas.

Fase de análisis

Para esta investigación se analizan variables de priorización, que se extraen directamente del conjunto de vulnerabilidades y data detectada en la fase de coleccion. El conjunto de datos recopilados permite entender y conocer variables de entorno de la organizacion en cuestión. Mediante este proceso se aborda la fase de analisis en la metodología OSINT, a continuación, se presentan las variables de entorno que son extraídas a partir de la data recopilada

Variables Globales

Este tipo de variables se generan a partir de la información que se extrae del conjunto total de vulnerabilidades halladas en las diferentes IPs.

Total Vulnerabilities (TV). Cualquier discusion pública sobre información acerca de vulnerabilidades puede ayudar a un pirata informático. Una organización necesita muchos recursos y trabajo para proteger sus redes y corregir todos los posibles agujeros. Mientras que para un pirata informático es mas fácil encontrar una sola vulnerabilidad, explotarla y comprometer la red. Alrededor del 52% de las vulnerabilidades explotadas son descubiertas por accion directa del ciberdelincuente (Alsowail & Al-Shehari, 2020), (IBM Corporation, 2020), (Vielberth, Bohm, Fichtinger, & Günther, 2020).

Cada vez son mas las herramientas de explotación de vulnerabilidades altamente sofisticadas que se hacen públicas. Por ejemplo, las herramientas de seguridad de la National Security Agency (NSA) filtradas en 2016 contenían cientos de exploits sofisticados y puertas traseras a sistemas de proveedores (Feutrill,

Ranathunga, Roughan, & Yarom, 2018). Al mismo tiempo, el parcheo o la actualizacion rápida de vulnerabilidades no es práctico en dominios como las redes de infraestructuras críticas debido a sus elevadas exigencias de disponibilidad.

Este panorama hace que el riesgo de una organización aumenta cuando se tiene gran cantidad de vulnerabilidades sin tratar. Al tomar en cuenta que el costo promedio global para las organizaciones por una brecha de datos y explotación de vulnerabilidades es de USD 86 millones (Alsowail R. A.-S., 2020), (IBM Corporation, 2020), se puede determinar que la cantidad de vulnerabilidades aumenta el riesgo de la organización, ya que la probabilidad de enfrentarse a los panoramas antes mencionados, aumenta. Por tal razón esta variable es la tasa de crecimiento o decrecimiento del riesgo general de una organización (Sharma & Singh, 2018), (CVE, 2021).

Shodan muestra la cantidad de vulnerabilidades identificadas en cada IP que rastrea. Por tal razon, en la Eq. 1, se presenta la siguiente sumatoria:

$$tv = \sum_{i=1}^{N} v_i$$

donde N es el número de IPs escaneadas, y v_i representa la cantidad de vulnerabilidades identificadas en la IP.

Average Organizational Risk (AOR). El sistema de puntuación CVSS es ampliamente investigado y usado en la mayoría de organizaciones a nivel mundial, al respetar el riesgo para cada vulnerabilidad (Shukla, Katt, & Obiora, 2019). Sin embargo, como se ha mencionado su puntuación está centrada en la vulnerabilidad. Por esta razón, las organizaciones emplean sus propios criterios adicionales de

evaluación para poder definir la real afectación a su entorno (Amankwan, Chen, Kwaku, Korkor, & Adutwun, 2020), (A. Dobrovoljc, 2017), (Keramati, 2016).

Dado que CVSS presenta un sistema de puntuacion cuantitativa, se pueden agrupar las vulnerabilidades de acuerdo a su rango de criticidad como se muestra en la Table 5. Ademas, se puede obtener un valor promedio del riesgo al cual está expuesta la organización ya que todas las vulnerabilidades identificadas estan inmersas en el mismo entorno. A través de la Eq. 2 se logra obtener el puntaje que demuestra el riesgo organizacional con base en las vulnerabilidades contenidas.

Tabla 5

Correspondencia entre la puntuación CVSS y el valor cualitativo (gravedad)

Score	Severity
0	Null
0.1 - 3.9	Low
4.0 – 6.9	Medium
7.0 – 8.9	High
9.0 – 10.0	Critical

Nota. La tabla representa las variables cualitativas que se asignan a los valores cuantitativos de una vulnerabilidad.

En la Eq. 2 se calcula el riesgo promedio de la organización definido por CVSS:

$$aor = \frac{\sum_{i=1}^{N} \sum_{j=1}^{vi} (CV\ SS_{ij})}{tv}$$

donde N representa el número de IPs de la organización, vi es la cantidad de vulnerabilidades contenidas en cada IP, por lo cual es necesario sumar cada CV SSij que contiene la vulnerabilidad. El valor AOR resultante, se relaciona con los rangos presentados en la Tabla 5 para medir la severidad del riesgo organizacional.

Average Vulnerability Time (AVT). El tiempo de remediación tiene en cuenta hasta que punto una organización está preparada para hacer frente a una vulnerabilidad. El nivel de preparacion de una organización frente a una vulnerabilidad puede afectar significativamente a la gravedad asociada a la misma y, por tanto, constituye una importante variable para la priorizacion de vulnerabilidades (Sharma & Singh, 2018).

Es importante considerar que cuando se publica la vulnerabilidad, generalmente no tiene aun, un parche de seguridad. Por esta razón, la puntuación de gravedad de una vulnerabilidad se ajusta a la baja sugiriendo un nivel de urgencia decreciente a medida que la reparacion se convierte, en definitiva. Cuanto menos oficial sea la correccion, mayor será la puntuación de la vulnerabilidad (CVE, 2021). Sin embargo, esta lógica permite definir que, si esa reparación no es aplicada en las organizaciones el riesgo de ese entorno se mantendrá ajustado al alta, debido a que el panorama inicial de afectación se mantiene.

El tiempo promedio de explotacion de una vulnerabilidad no parcheada en los sistemas estan disminuyendo rápidamente, de acuerdo con los informes presentados por las diferentes industrias les toma alrededor de 15 días explotar una vulnerabilidad a los ciberdelincuentes desde su descubrimiento (CISA, 2021). Aquellas organizaciones que estan en la mira de los ciberdelincuentes requieren un mayor esfuerzo para corregir los vectores de ataque. Por este motivo no se recomienda tener expuesta una vulnerabilidad tanto tiempo.

El tiempo promedio que le toma a una organizacion detectar y remediar una vulnerabilidad fue de 180 días en 2018 y para 2020 fue de 280 días, lo cual indica que va en aumento (IBM Corporation, 2020) , (Vielberth M. B., 2020). Por otro lado, las vulnerabilidades en su CVE ID ofrecen informacion referente al año en el cual se hicieron publicas, esto permite identificar hasta que punto la organización hace frente

a las vulnerabilidades conocidas. Independientemente del año en que se haya implementado una determinada tecnología, el CVE-ID demostrara que dicho servicio o software es vulnerable desde cierto tiempo atras, por lo cual es recomendable actualizarlo, si no se hace caso a las buenas practicas de seguridad la probabilidad de sufrir un ataque aumenta (CISA, 2021). Por tal razón, esta variable de tiempo es importante en el proceso de priorizacion.

En la Eq. 3 se calcula el tiempo promedio de las vulnerabilidades:

$$avt = \frac{\sum_{i=1}^{N} \sum_{j=1}^{vi} \left(Current_{year} - CVE_{year_{ij}} \right) \times 365)}{tv}$$

Dado que el valor de AVT resultante, corresponde a un promedio en días, se debe ajustar a un rango de valores entre 0 ≤ avt ≤ 1 para poder evaluar la probabilidad, al tomar como referencia el tiempo promedio de detección y remediación para las organizaciones descrito anteriormente se propone las siguientes condiciones:

- avt ≥ 280 → avt = 1
- 140 < avt < 280 → avt = 0.5
- avt ≤ 140 → avt = 0.1

Variables de IP

Estas tipo de variables se generan a partir de la información que se obtiene solamente de la IP escaneada y afecta directamente al conjunto de vulnerabilidades que se han identificado dentro de la IP.

Probability of Occurrence of an Event in the IP (POE). Dado que cada IP en la red es independiente (K. A. Farris, 2018) y asumiendo que la probabilidad de ocurrencia de un evento es independiente de la IP, se define la Eq. 4.

$$poe_{ip} = \frac{vip}{tv}$$

Donde, vip representa la cantidad de vulnerabilidades en la IP, esto define que a mayor cantidad de vulnerabilidades mayor es la probabilidad de ocurrencia de un evento, independientemente del impacto que este represente.

Probability of Open Ports (POP). La proteccion de la información y la alta disponibilidad de servicios requiere un gran esfuerzo técnico y tecnológico. Perder o exponer información o dejar servicios inactivos tiene un impacto negativo en las organizaciones, tanto a nivel funcional como reputacional (Al-dhaqm, Abd Razak, & Richard, 2020). La rapida transición digital deja expuestas vulnerabilidades que estan siendo explotadas por los ciberdelincuentes (Erza Aminanto, Ban, Isawa, Takahashi, & Inoue, 2020) , (Ron M. B., Fuertes, Bonilla, & Toulkeridis, 2018). Entre los incidentes de seguridad mas comunes están la infección por malware, el ransomware, el aprovechamiento de vulnerabilidades, el acceso indebido a aplicaciones, los ataques de ingeniería social y la denegacion de servicios (Goel & Nussbaum, 2021), (Sun, y otros, 2019).

Los puertos representan vectores de intercambio de informacion y comunicación, es decir sirven para identificar el proceso al que entregar el mensaje dentro de la maquina. Por esta razón en aquellos que están abiertos y expuestos directamente para el publico en internet representa un mayor riesgo y por tanto una prioridad mayor de atencion y control.

Shodan detecta los diferentes puertos abiertos que tiene la IP, por tal motivo se puede definir que estos puertos permiten el intercambio de información. Para poder obtener un valor cuantitativo se definen las siguientes ecuaciones:

$$top = \sum_{i=1}^{N} op_i$$

donde op es la cantidad de puertos abiertos en una IP, por tanto top representa el total de puertos abiertos en la organizacion con N IPs. Dado que cada IP es independiente en la red, se define el riesgo de puertos abiertos para cada IP a traves de la Eq 6:

$$pop_{ip} = \frac{op_{ip}}{top}$$

Probability for Query Tags (PQT). Durante el proceso de recopilación de direcciones IPs de la organización objetivo de estudio, se evidenció que Shodan devuelve una variable denominada tags. Esta variable contiene cadenas de caracteres que hacen referencia al servicio encontrado o alojado en la IP escaneada. Para las IPs objetivo de este estudio se evidencio tags como: ”database”, ”self-signed”. Se ha tomado como variable de priorizacion, debido a que cuando en una IP se detecta un tag, la variable ”data” de Shodan, contiene mayor cantidad de información específica sobre el servicio, por ejemplo, versiones, tecnologías, Sistemas Operativos, entre otras caracter´ısticas que en esa IP se alojan.

En la figura 8 se muestra la tendencia en las IP que presentan esta variable. Cuando las IPs tienen la variable tag, muestras mayor cantidad de data items, ademas el número de vulnerabilidades es mayor. Esto se debe a que Shodan conoce mayor data específica de la organización y por tanto puede relacionar mayor cantidad de vulnerabilidades conocidas.

Dado que la calidad de informacion expuesta en Shodan aporta mayor valor en el conocimiento e investigacion para los ciberdelincuentes (Lee, Shin, & Roh, 2017), (Zolotykh, 2021) , el riesgo aumenta cuando esta variable denominada tags tiene contenido alguno. Debido a que Shodan es un software cerrado y en su

documentacion tan solo se describe a la variable tag para los procesos de consulta y no en los procesos de respuesta, limitandose a verificar su existencia para poder asignar un valor de riesgo para el proceso de priorizacion (Shodan, 2021). Para poder obtener valores cuantitativos se propone las siguientes condiciones:

- IF (tags > 0) → pqt = 1
- IF (tags = 0) → pqt = 0

Figura 8

Número de datos y vulnerabilidades cuando se muestra la etiqueta de consulta

Nota. La figura representa una comparativa de la informacion que muestra Shodan entre las IP que contienen o no la variable tags.

Variables de Vulnerabilidad

Este tipo de variables se generan a partir de la información propia de cada vulnerabilidad y afecta solamente a la vulnerabilidad en cuestión.

Total References (TR). Cada registro CVE incluye las referencias donde se puede entender un contexto mas amplio acerca de la vulnerabilidad. Las referencias deben apuntar al contenido que es relevante para la vulnerabilidad e incluir al menos todos los detalles incluidos en el registro CVE. Una característica fundamental es que las referencias tambien deben estar disponibles publicamente (CVE, 2021).

Existen dos enfoques en esta variable. Por un lado, las referencias son fuentes de informacion valiosa al momento de proceder con la corrección o mitigación de una vulnerabilidad, y por otro lado esa misma información permite del lado del atacante conocer de primera mano que productos y versiones estan siendo afectas. En varios casos se ha identificado también que presenta los exploits con los cuales se aprovechan dichas vulnerabilidades (Alperin, Wollaber, Ross, Trepagnier, & Leonard, 2019), (Sharma & Singh, 2018).

Para el caso de estudio se emplea el segundo enfoque, principalmente la razón es que al aplicar técnicas OSINT lo que se propone es obtener la mayor cantidad de información que permita comprometer la integridad, disponibilidad y confidencialidad de la organizacion. Además, se mantiene el punto de vista del atacante, donde toda esta información está accesible sin limitacion alguna, y le permite realizar un proceso de inteligencia para entender la infraestructura tecnológica. Con base en el enfoque propuesto, y al considerar que el número promedio de referencias de una vulnerabilidad con amplio entendimiento es mayor a 8 (Haipeng, Liu, Liu, Park, & Subrahmanian, 2019), se proponen las siguientes condiciones:

- IF (Vreferences ≥ 10) → tr = 1
- IF (Vreferences < 10) → tr = $(\frac{\text{Vreferences}}{10})$

Exploitation Probability (EP). Para los procesos de identificación y detección de vulnerabilidades los investigadores hacen uso de exploits, que son codigos que muestran la existencia de una falla, es decir, se puede comprometer la confidencialidad, integridad o disponibilidad. Una puntuación CVSS es indicativa de la gravedad de la vulnerabilidad, pero no ayuda a predecir la demora del exploit (Feutrill, Ranathunga, Roughan, & Yarom, 2018).

Se ha demostrado que mientras el 94% de las vulnerabilidades explotadas ten´ıan un exploit disponible en 30 días, solo el 72% de los parches estaban disponibles en ese plazo (Frei, 2009), lo cual es un indicador que en su gran mayoría para las vulnerabilidades antiguas existe un exploit disponible. Las puntuaciones CVSS no permiten discriminar eficazmente entre la probabilidad de explotacion y de no explotación (Allodi & Massacci, 2012). Sin embargo, hay estudios que han demostrado que las vulnerabilidades de alto riesgo tienen una mayor probabilidad de ser explotadas (Feutrill, Ranathunga, Roughan, & Yarom, 2018), (Rajasooriya, Tsokos, & Kaluarachchi, 2017), en otros estudios se le conoce tambien a esta variable como la edad de la vulnerabilidad (K. A. Farris, 2018). En relación a este panorama ampliamente investigado se propone las siguientes condiciones para priorizar la atención de las vulnerabilidades:

Dado que $CVE_{year} \leq Current_{year}$ y considerar las estadísticas mostradas previamente, se propone lo siguiente:

- IF (CV SS ≥ 7.0) → ep = 1
- IF (4 ≤ CV SS ≤ 6.9) → ep = 0.5
- IF (0 ≤ CV SS ≤ 3.9) → ep = 0.1

Proceso de extracción de conocimientos

Los algoritmos, modelos y fórmulas matemáticas de priorización tienen muchos problemas cuando se aplican a un entorno del mundo real, por los múltiples objetivos y que constantemente son conflictivos en las organizaciones. Precisamente, los diferentes problemas tienen una enorme importancia práctica, ya que producen un conjunto de soluciones basadas en la importancia asignada a cada uno de los objetivos (K. A. Farris, 2018).

En esta investigación se propone un modelo de priorización basado en el conocimiento que ofrecen las propias vulnerabilidades y ciertas características donde se alojan, de acuerdo con la organizacion objetivo. Es decir, basado en el conjunto de vulnerabilidades detectadas, se obtienen las variables cuantitativas presentadas anteriormente, las cuales demuestran conocimiento acerca de la gestión y riesgo de la organización en cuestión. Para poder cuantificar cada vulnerabilidad, basada en las variables de entorno identificadas, se calcula el factor de riesgo que cada una representa para la organización, a continuación de describe el proceso de cálculo.

Risk Factor (RF). Este factor se define por 2 vectores la probabilidad de ocurrencia, y el impacto (Jeon & Kang, 2021), (Lee, Shin, & Roh, 2017). El *RF* viene determinado por la siguiente ecuación Eq. 7.:

$$RF = Probability\ of\ occurrence \times Impact$$

En este estudio se tiene 3 tipos de variables:

- Global Variables
- IP Variables
- Vulnerability Variables

Dado que las variables tienes diferente grado de afectación, es necesario correlacionarlas para determinar la probabilidad de ocurrencia. Al considerar que para este caso de estudio no se ha definido importancia o superioridad por el tipo de variable, es posible promediar los valores como se presenta en la Eq. 8.

Probability of occurrence:

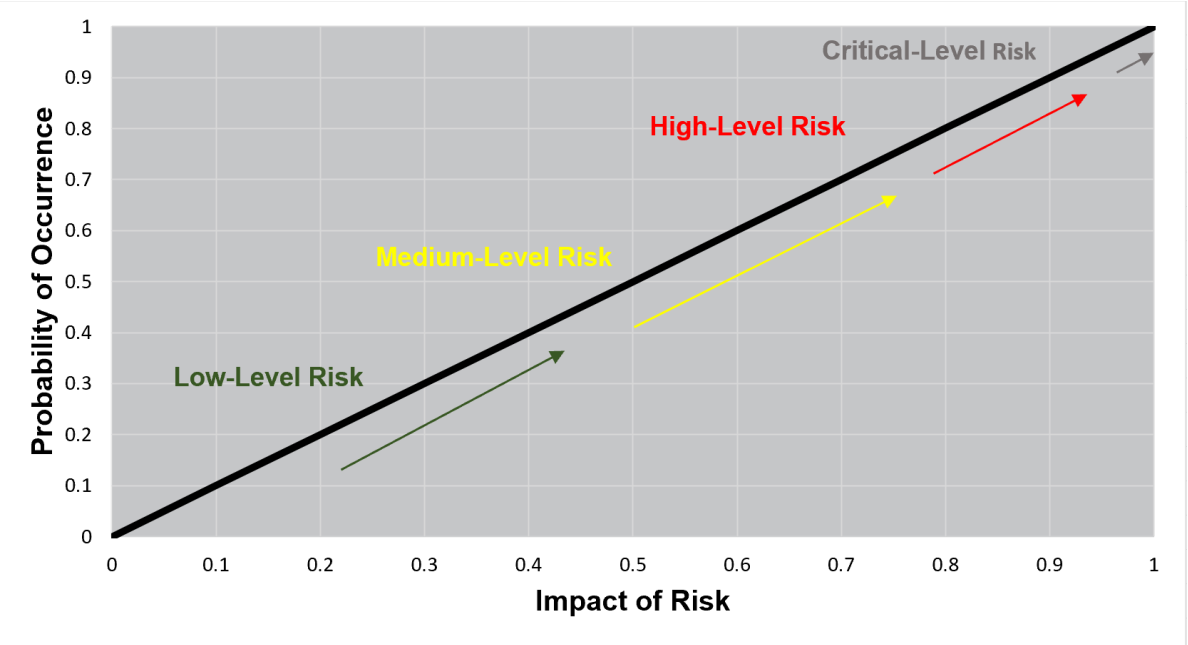
$$po = \left(\frac{tr + ep}{2}\right) \times \left(\frac{poe + pop + pqt}{3}\right) \times avt$$

El impacto es un parámetro ampliamente investigado y estudiado en el framework CVSS, donde se establecen varias métricas para evaluar las vulnerabilidades y como estas afectar´ıan a los elementos del entorno de seguridad IT si se llegan a materializar (INCIBE-CERT, 2015) (CISA, 2021). Al considerar que el objetivo de este estudio es apoyarse en la puntuacion CVSS para aproximar a un valor más exacto enfocado en el entorno de afectación, se determina la siguiente igualdad I = CV SS.

Finalmente los valores de priorización asignados a las vulnerabilidades luego del cálculo del RF, se agrupan de acuerdo a un rango de criticidad presentados en la tabla 5 . En la Figura 9 se presenta graficamente el RF por rangos.

Figura 9

Factor de riesgo



Nota. La figura representa el factor de riesgo de acuerdo con la probabilidad de ocurrencia de un evento y el impato de la vulnerabilidad.

Ejemplo ilustrativo. Para cumplir con esta fase, es necesario obtener el resultado y conocimiento. Luego de haber procesado la información, para el caso de estudio el Factor de Riesgo representa el conocimiento. Cuando se aplica este proceso al conjunto total de vulnerabilidades, se logra una priorización cuantitativa que se aproxima a la realidad organizacional debido a la incorporación de las variables de entorno. Para fines ilustrativos se detalla el cálculo en una vulnerabilidad espec ífica, para este caso se toma la vulnerabilidad CVE-2017-9798 que se detalló anteriormente. La organizacion que fue objeto de este estudio presentó las siguientes variables:

- Global Variables
    - tv = 541
    - avt = 1
- Ip Variables
    - poe = $(\frac{96}{541})$ = 0.18
    - pop = $(\frac{9}{14})$ = 0.64
    - pqt = 1
- Vulnerability Variables
    - tr = 1
    - ep = 0.5
    - CV SS = $\frac{5}{10}$ = 0.5

$$RF = \left(\frac{1 + 0.5}{2}\right) \times \left(\frac{0.18 + 0.64 + 1}{3}\right) \times 1 \times 0.5$$

$$RF = 0.23$$

Finalmente, al tomar como parámetros de evaluación la información disponible para el atacante el RF de la vulnerabilidad antes presentada disminuye del valor referente que CVSS que se le ha asignado. Se abordará a detalle este panorama en la sección de resultados.

Capítulo IV

Desarrollo del prototipo de detección y priorización de vulnerabilidades

La cantidad de vulnerabilidades que se detectan en las organizaciones, cada vez van en aumento. La rapida transicion digital al que se enfrentan, obliga al personal a realizar configuraciones e implementaciones apresuradas, las cuales son sujetas a pruebas y testeos continuos. Estos procesos son constantes motivo por el cual las vulnerabilidades aparecen y desaparecen con cada fase de trabajo.

El proceso de escaneo, detección, y priorización de vulnerabilidades debe ser continuo y automatizado ya que las variables de entorno cambian de acuerdo a los valores que se obtengan. Los procesos de mitigación y tratamiento de vulnerabilidades hacen que las infraestructuras tecnologicas mejoren, o mantengan sus niveles de seguridad. Sin duda, en entornos tan cambiantes, es un gran reto para las organizaciones mantener estos procesos.

Al considerar que no existen metodologias ni procesos de desarrollo definido para este tipo de sistemas, las metodologias que se apliquen para el proceso de desarrollo deben permitir mostrar resultados rápidos, cambios y evaluaciones constantes durante su construcción, ya que al ser un proceso complejo de tratamiento de datos es importante buscar resultados cada vez mas eficientes. Por otro lado, de los estudios analizados se concluyó que el 48% aplican prototipos para evaluar los procesos propuestos, esto debido a que reduce costos y tiempo, ademas se evaluan los resultados de forma continua y se ajustan con los cambios requeridos. En la figura 6 se mostró las fases propuestas en la investigación, y las cuales se seguirán para el desarrollo del prototipo.

Definición del enfoque de detección de vulnerabilidades

En la sección Metodología multifactorial de recolección, análisis y toma de decisionesse se presentó de manera detallada el enfoque de deteccion para este estudio. Sin embargo, es importante considerar que de acuerdo a la clasificacion realizada en nuestro estudio previo, se basa principalmente en la deteccion de vulnerabilidades conocidas de software, pero tambien se obtiene informacion relevante sobre vulnerabilidades a nivel de Red. Debido a que Shodan brinda una amplia información el enfoque para esta investigación se considera mixto, ya que se hace el uso de los dos enfoques y sus variables como se muestra en la Tabla 6 para poder priorizar el orden de atención de las vulnerabilidades detectadas.

Tabla 6

Variables del enfoque de detección de vulnerabilidades

Software	CVE
	Software data
Network	IP
	Open Ports
	Query Tags
	Domains
	Hostnames
	Organization Name

Nota. La tabla presenta las variables por cada VDA

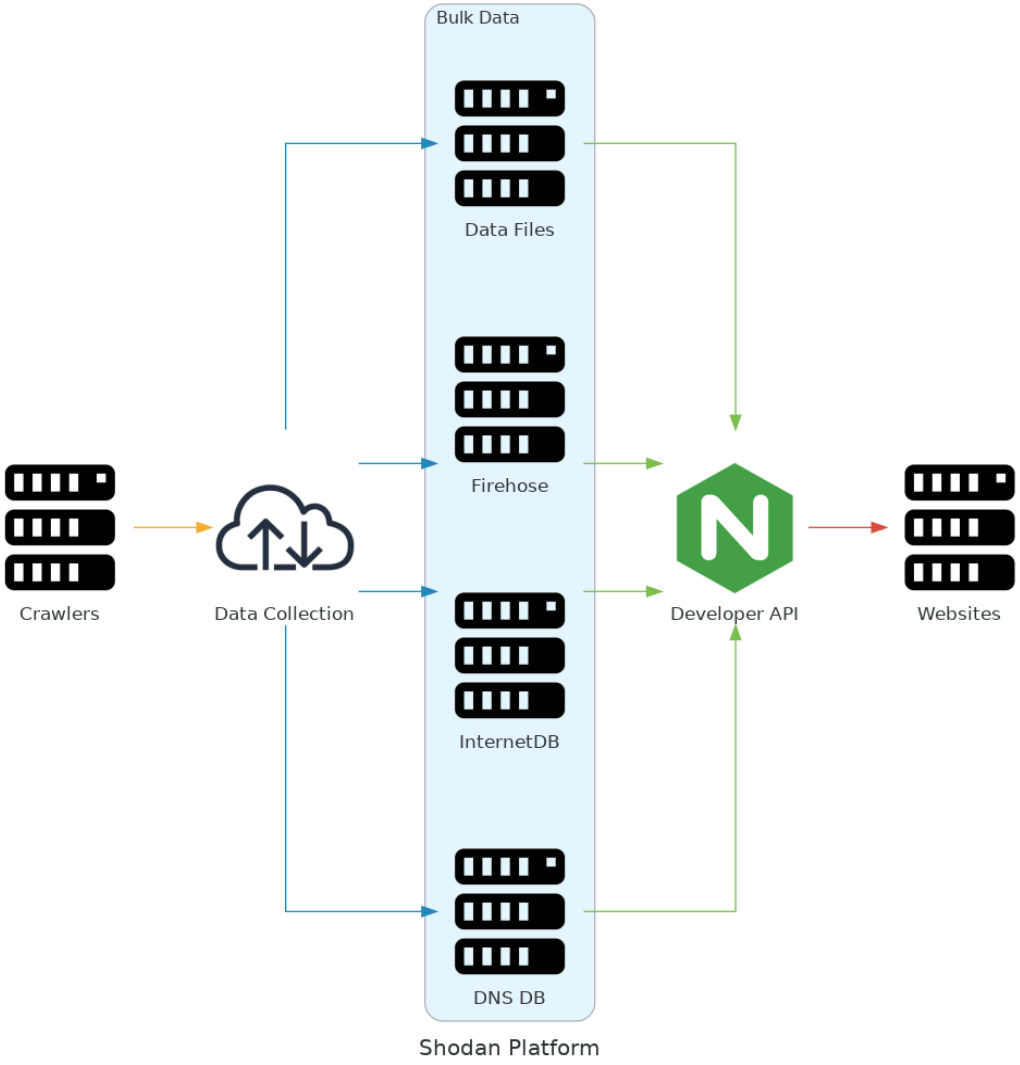
Diseño Arquitectura del sistema

Shodan a través de su documentación menciona que todos sus sitios web se construyen completamente sobre la misma API pública de Shodan (Shodan, 2021). Esto significa que cualquier cosa que se pueda hacer a traves del sitio web támbien se puede hacer mediante programación al usar las API.

En la Figura 10 se muestra un diagrama de alto nivel que muestra el flujo de datos en la plataforma Shodan. De acuerdo con el concepto de funcionamiento de los websites de Shodan, se puede observar que Shodan como tal es un servidor donde se almacenan los datos que han sido hallados por sus crawlers. De este modo, los datos ya presentan un preprocesamiento y mapeo. Sin embargo, aún es información muy dispersa por lo cual en el sitio web oficial no se muestra toda la información que la API devuelve.

Figura 10

Diagrama de alto nivel que muestra el flujo de datos en la plataforma Shodan



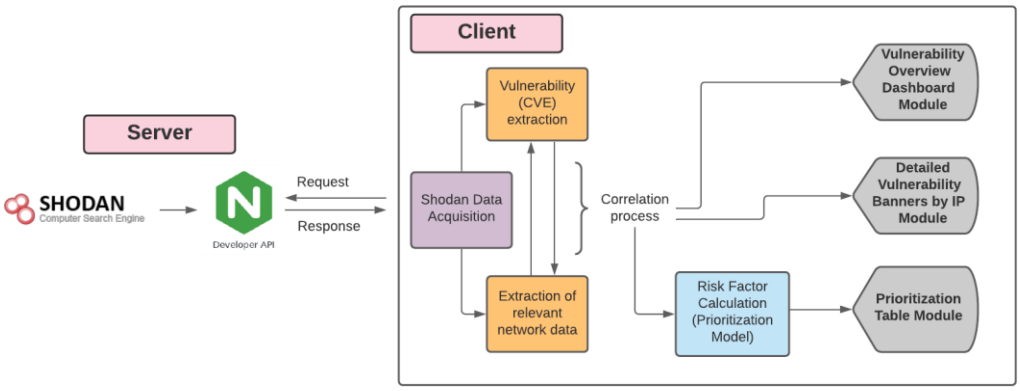
Nota. La figura representa la estructura de funcionamiento de Shodan. Tomado de *Shodan*. 2021 (septiembre de 07 de 2021). Obtenido de https://www.shodan.io.

En la Figura 11, se propone una arquitectura Cliente – Servidor, donde se realizó la consulta a Shodan para obtener la información correspondiente a las IPs de una misma organización, la cual entra en un proceso de extracción y análisis de datos los cuales son presentados en un servicio web a través de 3 módulos que muestran la información correlacionada acerca de las vulnerabilidades detectadas. El objetivo de estos módulos es brindar conocimiento al cliente acerca de una organización especifica. La correlación de vulnerabilidades y características de red, evitan la dispersión de datos que por si solos no tienen relevancia y requieren del capital

humano para ser interpretados y analizados. Se logró optimizar recursos y sobre todo tiempo al momento de conocer, interpretar y priorizar las vulnerabilidades que Shodan ha detectado.

Figura 11

Arquitectura del consumo de recursos del prototipo y diagrama de procesamiento de datos



Nota. La figura representa el proceso de consulta y tratamiento de datos a través del prototipo.

Definición la metodología de desarrollo

Modelo de prototipos

En Ingeniería de software, pertenece a los modelos de desarrollo evolutivo. De acuerdo con la características de los estudios analizados se ajusta a esta investigacion para realizar pruebas rapidas que permitan evaluar los resultados y corregir errores (Vielberth, Bohm, Fichtinger, & Günther, 2020). El prototipo debe ser construido en poco tiempo, al usar los programas adecuados y no se debe utilizar muchos recursos.

Al considerar que el core de esta investigación es el tratamiento y presentación de datos, el desarrollo se centra en una representación de aquellos módulos que serán

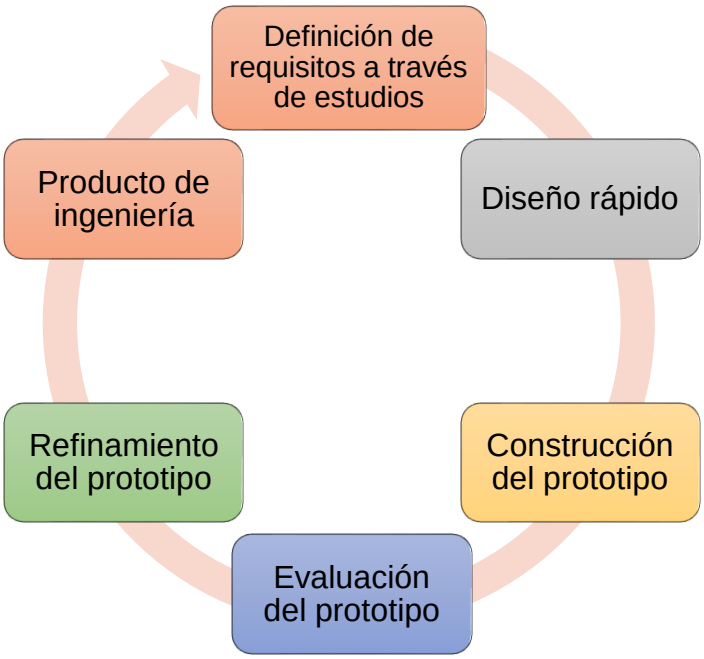
visibles para la interpretación de la información (Sharma , Sibal, & Sabharwal, 2020). El diseño que conduce a la construcción de un prototipo, debe ser sujeto a una retroalimentación que ayude a la refinación continua hasta llegar a un punto satisfactorio de conformidad; gracias a esto se refinan los requisitos del software final. El objetivo de este modelo es que los resultados sean visibles a corto plazo.

Para este tipo de sistemas este modelo es útil cuando se conoce los objetivos generales para el software, pero no identifica los requisitos detallados de entrada, procesamiento o salida. Por tal razón el prototipo permite entender esos valores y ajustarlos de acuerdo con los criterios mínimos de evaluación de vulnerabilidades (Zhu, 2020). Además, ofrece un mejor enfoque cuando el responsable del desarrollo del software está inseguro de la eficacia de un algoritmo, de la adaptabilidad de un sistema operativo o de la forma que debería tomar la interacción humano-máquina.

De acuerdo con las prestaciones de este modelo, se ha identificado que se ajusta a las necesidades de esta investigación, ya que al ser un panorama incierto es beneficioso realizar pruebas rápidas sin ocupar gran cantidad de recursos para sacar conclusiones y afinar los procesos de detección, correlación y priorización de vulnerabilidades.

Figura 12

Modelo de Prototipos



Nota. La figura muestra el proceso de desarrollo a través del modelo de prototipos. Adaptado de *Assessment of vulnerability*, por G. Spanos, 2017.

Desarrollo del prototipo

El prototipo fue desarrollado en React ya que es una biblioteca Javascript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página. Es mantenido por Facebook y la comunidad de software libre. La facilidad de interactuar con código frontend y backend, permite realizar pruebas dinámicas rápidas. Shodan devuelve la información en formato JSON, por lo cual es necesario un proceso de mapeo para su presentación e interpretación. Dado que esta investigación se ha centrado en la extracción de vulnerabilidades y de acuerdo con la arquitectura presentada en la Figura 11, se determina como un proceso la extracción solo de vulnerabilidades de toda la data, se

desarrolló el algoritmo.1, donde se extrae toda la información relacionada con cada CVE dentro de la variable “data” de Shodan que es la que contiene a detalle cada CVE.

Algoritmo 1

Extracción de vulnerabilidades

Algorithm 1: Extract Vulnerabilities

```
Data: Variable data and vulns of Shodan
Result: Vulnerabilities with description, CVSS and references
        (returnData)
tempValue ← Array[empty];
returnData ← Array[empty];
count ← 0;
for i ← 0 , data.length do
    if data[i].vulns ≠ undefined then
        tempValue.push(data[i].vulns);
    end
end
for i ← 0 , data.length do
    count ← 0;
    for j ← 0 , tempValue.length do
        if tempValue[j][vulns[i]] ≠ undefined then
            if vulns[i].index = tempValue[j][vulns[i]].index and
            count < 1 then
                tempValue[j][vulns[i]].cve ← vulns[i];
                returnData.push(tempValue[j][vulns[i]]);
                count ++;
            end
        end
    end

end
```

Una vez que se ha definido la funcion de extracción de vulnerabilidades, se hace llamado a la función que se encarga de conectar con Shodan. El algortimo.2 se encarga de conectar, mapear y extraer la información, como se puede notar en esta función se hace llamado al algoritmo.1, el objetivo de este proceso es que una vez

que Shodan devuelva la información se pueda separar y devolver dos valores que contendran solo la información necesaria de vulnerabilidades y de red, lo cual facilita el tratamiento y correlación para los procesos de priorización y presentación en pantalla.

Algoritmo 2

Adquisición de data de Shodan

Algorithm 2: Shodan Data Acquisition

Data: $shodanData$ in JSON format from
https://api.shodan.io/shodan/host/ip?key=API_KEY where ip
is input variable
Result: Network and Vulnerabilities values ($\{network, vuln\}$)
$vuln \leftarrow Array[empty]$;
$network \leftarrow Array[empty]$;
if *shodanData.vulns* $\neq$ *undefined* **then**
    network.push(shodanData);
    vuln.push(extractVulnerabilities(shodanData.data ,
     shodanData.vulns));
end

Por último, para el proceso de priorización, se transformó en algoritmos las fórmulas presentadas en la metodología de la investigacion, aplicar estructuras de control de flujo cíclica para calcular los valores necesarios y asignarlos como variables a cada vulnerabilidad. El algoritmo.3 se encarga de asignar el factor de riesgo a cada vulnerabilidad para finalmente ordenarlas para una priorización basado en la información que se ha detectado disponible para el atacante.

Algoritmo 3

Cálculo del factor de riesgo

Algorithm 3: Risk Factor Calculation

Data: $info = \{network, vuln\}$ with tr, ep, poe, pop, pqt and avt values

Result: $vulnerabilities$ with risk factor value $(info)$

```
for i ← 0 , info.length do
    for j ← 0 , info.vuln.length do
        info[i].vuln[j].po ← ((info[i].vuln[j].tr + info[i].vuln[j].ep)/2)
          * ((info[i].network.poe + info[i].network.pop +
          info[i].network.pqt)/3) * avt;
        info[i].vuln[j].impact ← info[i].vuln[j].cvss /10;
        info[i].vuln[j].rf ← info[i].vuln[j].po * info[i].vuln[j].impact;
    end
end
```

Validación de la Operación del Prototipo

En esta fase se describe los datos de entrada para comprobar el funcionamiento del prototipo, ademas se analizan los resultados obtenidos en el proceso de priorización de acuerdo con el modelo propuesto.

En esta invetigación se propone un modelo genérico que puede ser aplicado a cualquier organización, por lo cual es suficiente conocer un grupo de direcciones IP que sean parte de la infraestructura tecnológica para comenzar con la detección y análisis de vulnerabilidades. Para este caso de estudio se han descubierto 9 direcciones IP presentadas en la tabla 7, las cuales contienen vulnerabilidades y pertenecen a la Universidad de las Fuerzas Armadas ESPE. De acuerdo con esta sección se va a presentar los 3 módulos resultantes y descritos en la Figura 11.

Tabla 7

IP para el proceso de escaneo

IP
192.188.58.61
192.188.58.50
192.188.58.45
192.188.58.59
192.188.58.63
192.188.58.78
192.188.58.76
192.188.58.180
192.188.58.75

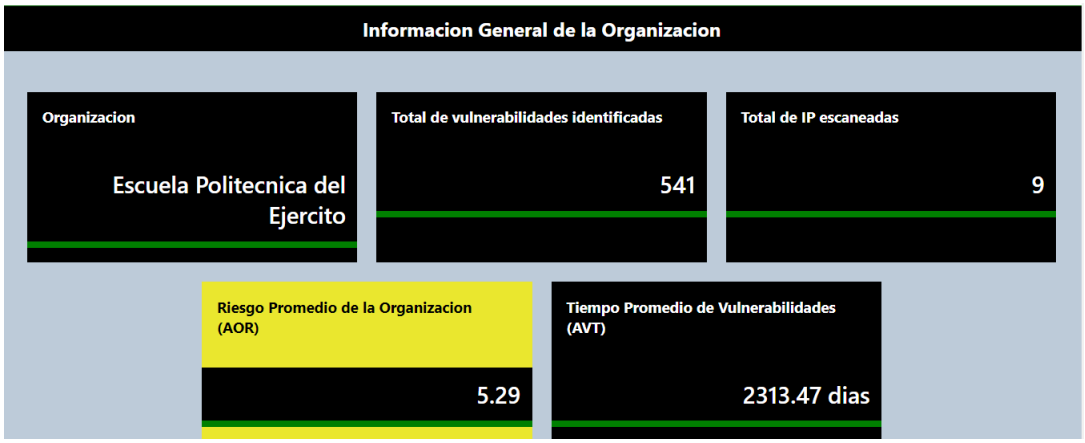
Nota. La tabla muestra las 9 IPs que se han seleccionado para el proceso de validación ya que se han identificado previamente vulnerabilidades

Módulo Dashboard General de Vulnerabilidades

Este módulo muestra una información general relacionada con la red y las vulnerabilidades detectadas, el objetivo de este módulo es brindar una visión global de la data resultante. De las 9 IPs escaneadas se han detectado 541 vulnerabilidades, las mismas que se verificaron en Shodan directamente a través de su website. El conjunto de vulnerabilidades muestra un riesgo organizacional de 5.29. De acuerdo con la tabla 5, corresponde a una severidad media, además el tiempo promedio de las vulnerabilidades identificadas supera los 2300 días. En la Figura 13 se presenta la información general de la organización.

Figura 13

Información General de la Organización

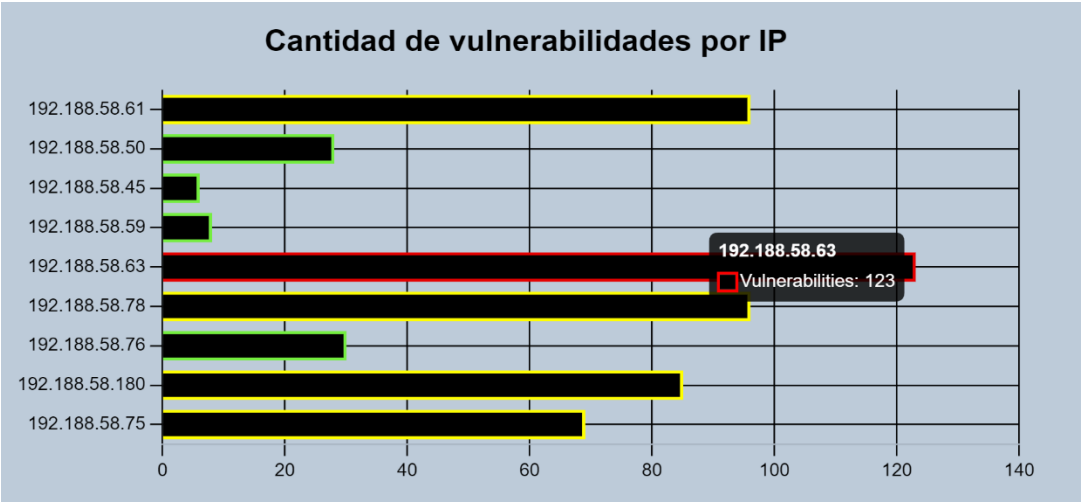


Nota. La figura presenta una captura de pantalla del prototipo relacionada con el módulo 1

En el mismo modulo se presenta la información correlacionada con cada IP de todas las vulnerabilidades que se identificaron en Shodan. A traves de cinco graficas se intenta mostrar el panorama general de la organización.

Figura 14

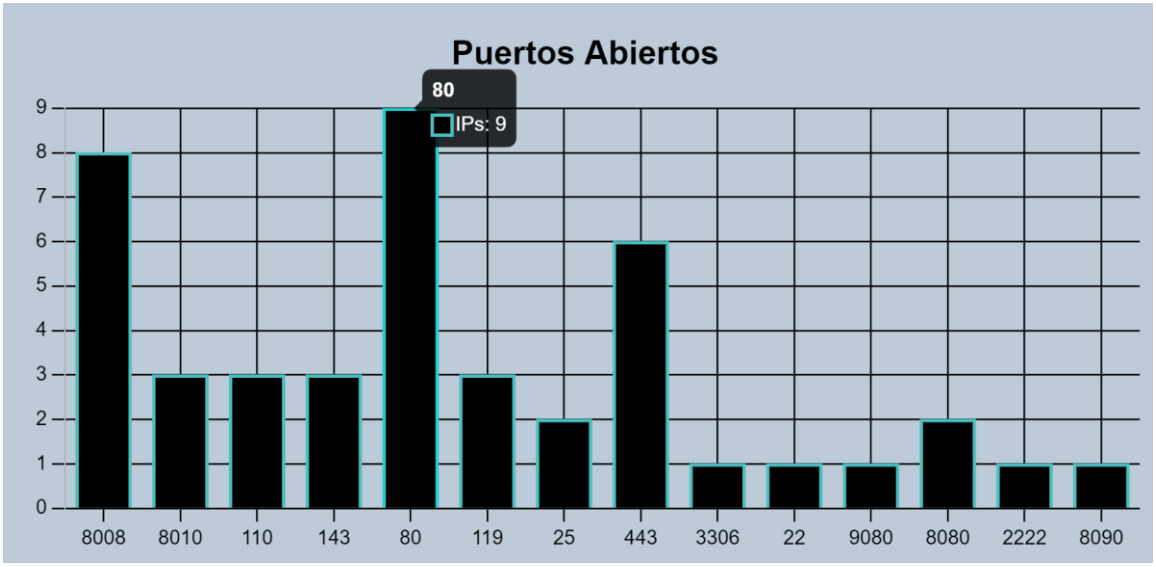
Cantidad de Vulnerabilidades por IP



Nota. La figura muestra un gráfico de barras para la cantidad de vulnerabilidades por IP.

Figura 15

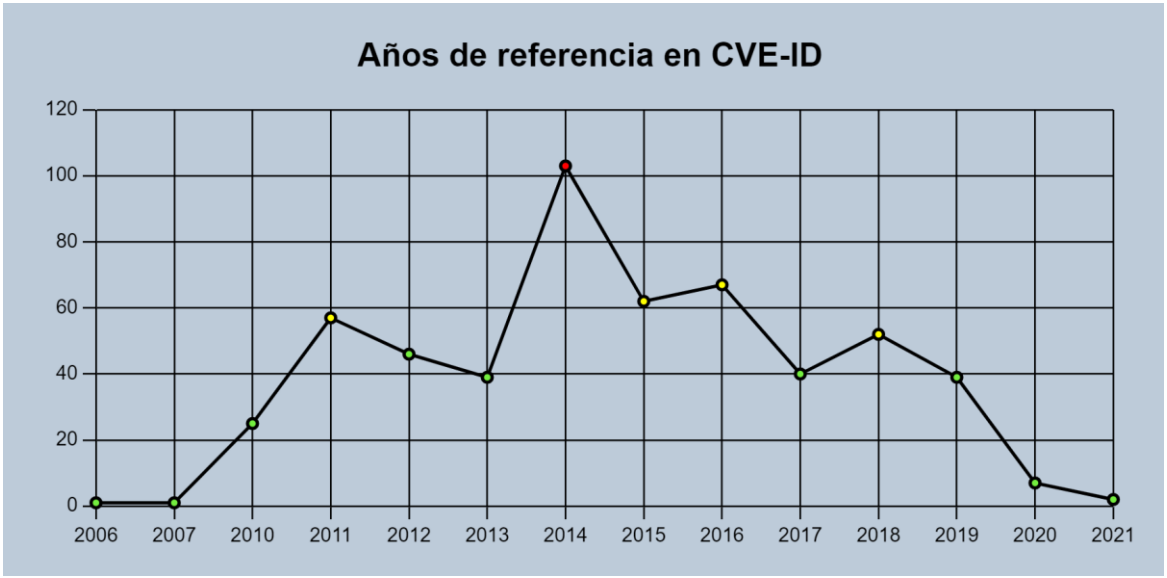
Cantidad de Puertos Abiertos



Nota. La figura muestra un gráfico de barras para la cantidad de puertos abiertos.

Figura 16

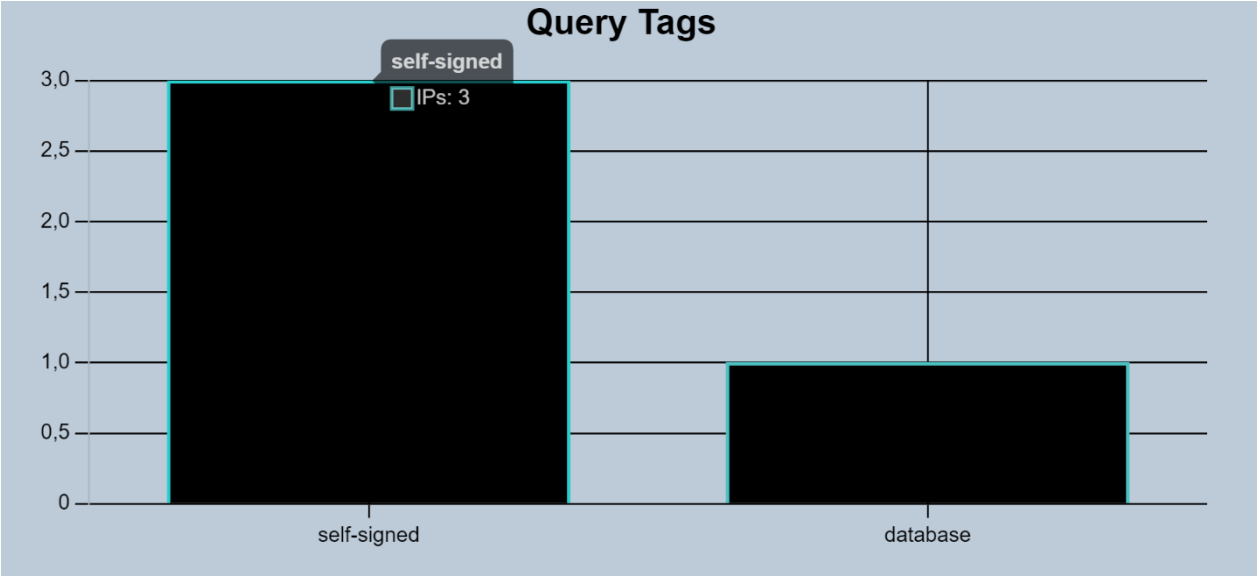
Años de referencia en el identificador CVE-ID



Nota. La figura muestra un gráfico de líneas de los años en que se hicieron públicas las vulnerabilidades identificadas.

Figura 17

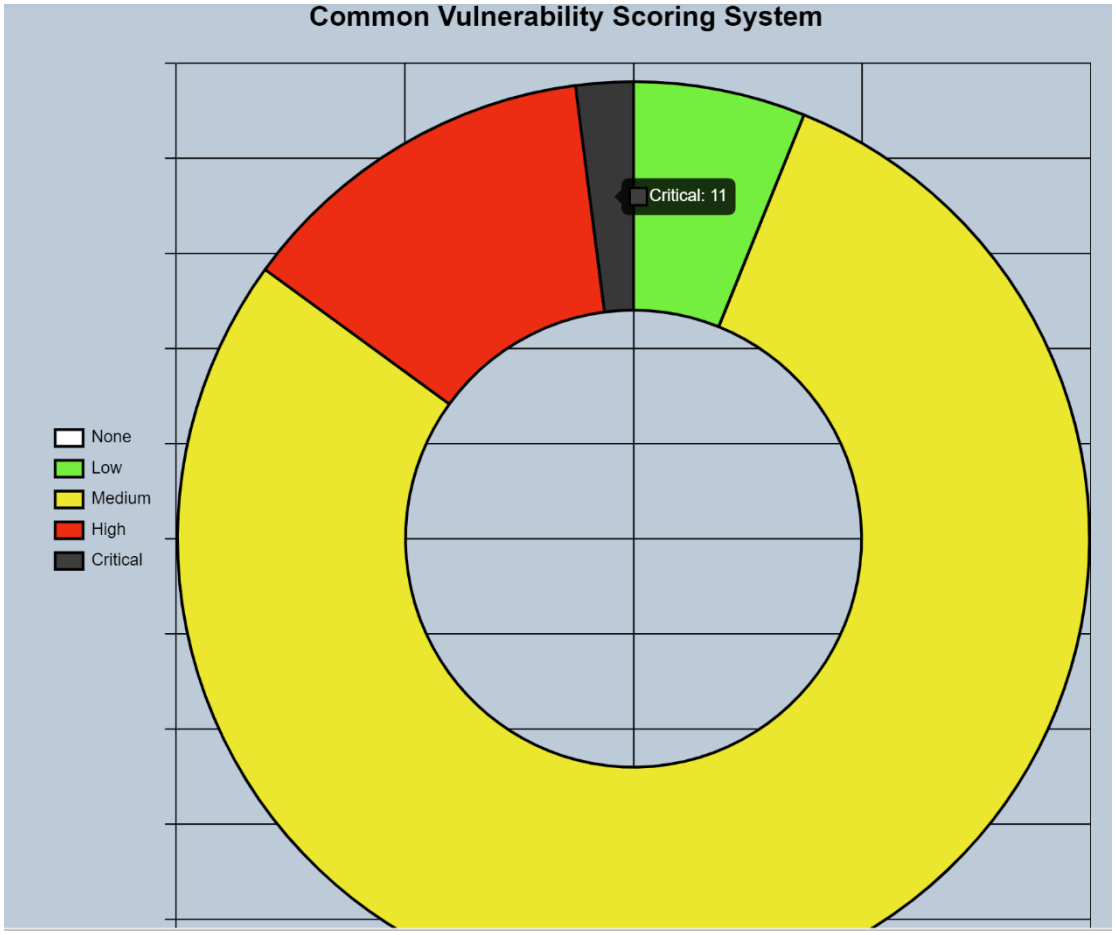
Cantidad de Query Tags



Nota. La figura muestra un gráfico de barras de acuerdo con los Query Tags identificados por Shodan.

Figura 18

Visión General de CVSS Score del total de vulnerabilidades



Nota. La figura muestra un gráfico circular con colores de riesgo el total de vulnerabilidades por cada rango CVSS.

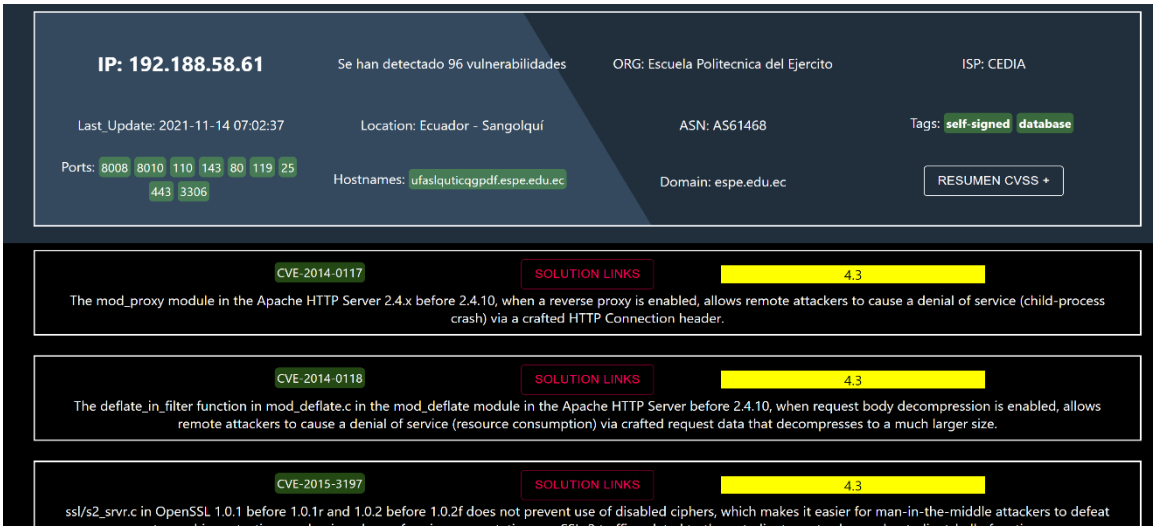
Banners Detallados de Vulnerabilidades por cada IP

Este módulo contiene a detalle la información relacionada con cada IP escaneada y las vulnerabilidades que en ella se han identificado. En la Figura 19 se presenta un extracto del módulo presentado en el prototipo. Este modulo contiene el detalle breve que la NIST le da a cada CVE con el objetivo de brindar un entendimiento a los analistas de ciberseguridad acerca del problema de la vulnerabilidad. Por otro lado en la Figura 20, 21 y 22 se muestra la posibilidad que nos da este modulo para poder ir a los enlaces externos para realizar un proceso investigativo acerca del

servicio que se aloja en la IP a traves de sus hostnames, puertos y acceder directamente a NVD para conocer a detalle la vulnerabilidad.

Figura 19

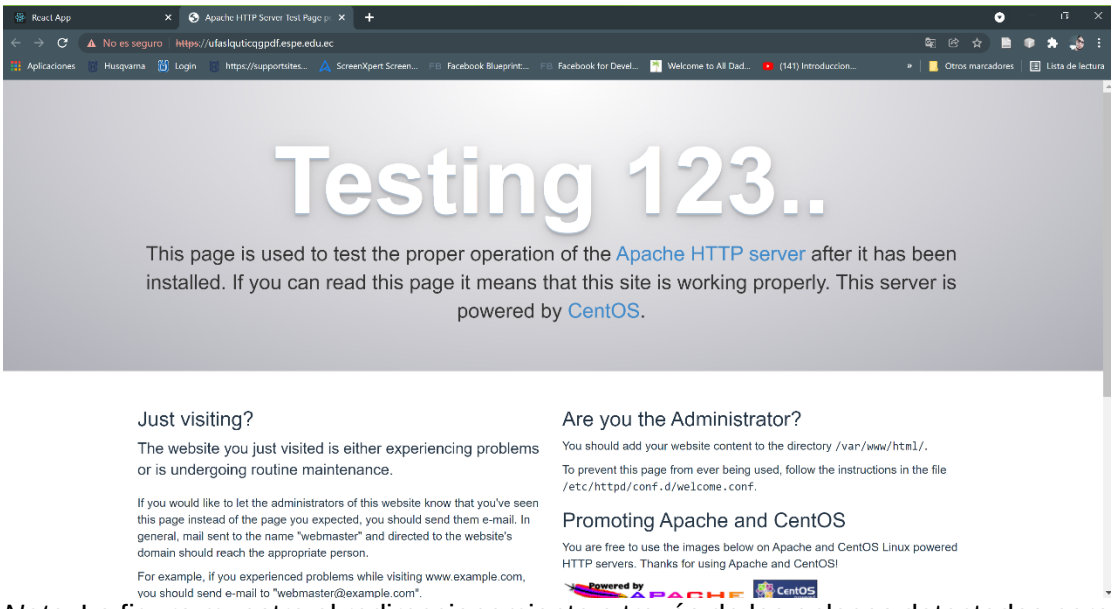
Banner de información detallada por cada IP



Nota. La figura muestra un extracto del prototipo donde se presenta a detalle las vulnerabilidades por cada IP

Figura 20

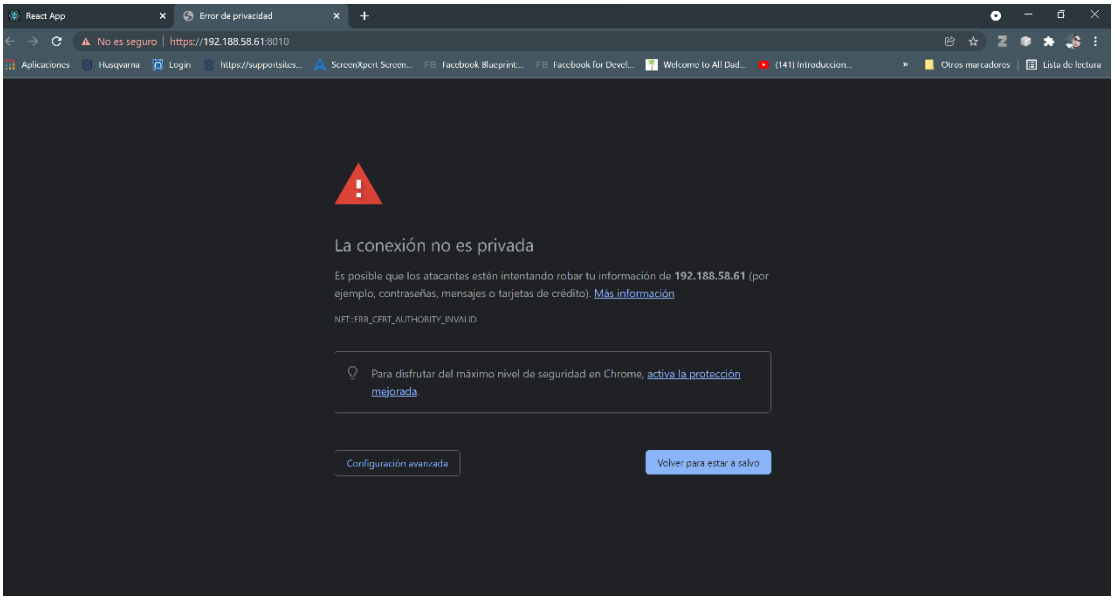
Enlace a los hostnames detectados



Nota. La figura muestra el redireccionamiento a través de los enlaces detectados para los hostnames.

Figura 21

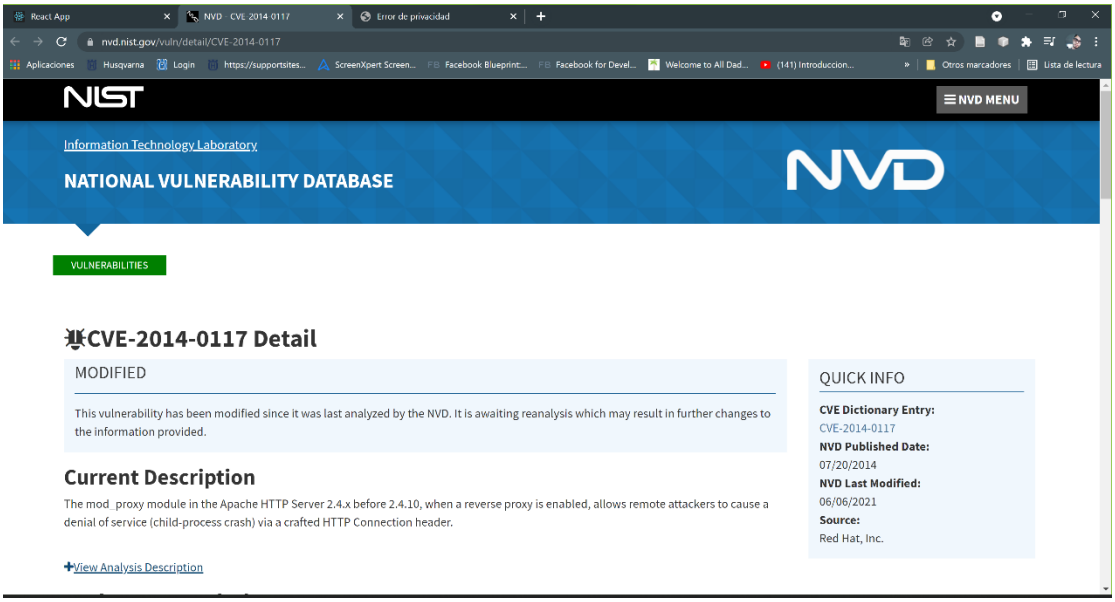
Enlace a los puertos detectados



Nota. La figura muestra el redireccionamiento a través de los enlaces detectados para los puertos.

Figura 22

Enlace a NVD



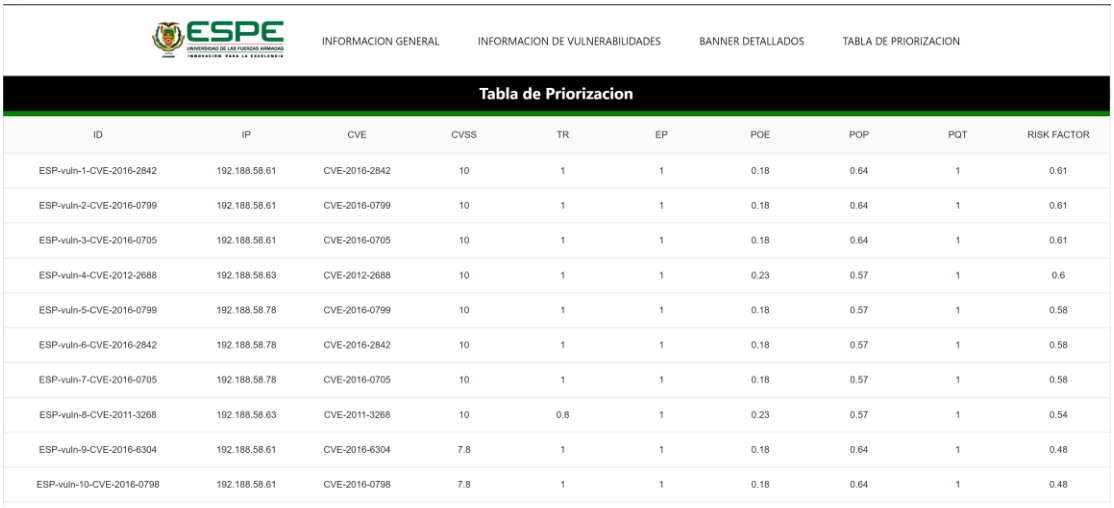
Nota. La figura muestra el redireccionamiento a través de los enlaces detectados para CVE.

Módulo de tabla de priorización

Este módulo contiene cada una de las vulnerabilidades y las variables de priorización calculadas. Se han ordenado de mayor a menor de acuerdo con el factor de riesgo para sugerir al usuario el orden de revisión. En la Figura 23 se presenta un extracto del módulo presentado en el prototipo.

Figura 23

Tabla de priorización de vulnerabilidades



ID	IP	CVE	CVSS	TR	EP	POE	POP	PQT	RISK FACTOR
ESP-vuln-1-CVE-2016-2842	192.188.58.61	CVE-2016-2842	10	1	1	0.18	0.64	1	0.61
ESP-vuln-2-CVE-2016-0799	192.188.58.61	CVE-2016-0799	10	1	1	0.18	0.64	1	0.61
ESP-vuln-3-CVE-2016-0705	192.188.58.61	CVE-2016-0705	10	1	1	0.18	0.64	1	0.61
ESP-vuln-4-CVE-2012-2688	192.188.58.63	CVE-2012-2688	10	1	1	0.23	0.57	1	0.6
ESP-vuln-5-CVE-2016-0799	192.188.58.78	CVE-2016-0799	10	1	1	0.18	0.57	1	0.58
ESP-vuln-6-CVE-2016-2842	192.188.58.78	CVE-2016-2842	10	1	1	0.18	0.57	1	0.58
ESP-vuln-7-CVE-2016-0705	192.188.58.78	CVE-2016-0705	10	1	1	0.18	0.57	1	0.58
ESP-vuln-8-CVE-2011-3268	192.188.58.63	CVE-2011-3268	10	0.8	1	0.23	0.57	1	0.54
ESP-vuln-9-CVE-2016-6304	192.188.58.61	CVE-2016-6304	7.8	1	1	0.18	0.64	1	0.48
ESP-vuln-10-CVE-2016-0798	192.188.58.61	CVE-2016-0798	7.8	1	1	0.18	0.64	1	0.48

Nota. La figura muestra la tabla de priorización generada a través del prototipo

Debido a que los datos se correlacionan entre si para lograr un valor resultante del factor de riesgo, es posible validar el valor obtenido para el ejemplo ilustrativo con el valor obtenido en el prototipo, se asegura el proceso de cálculo para el prototipo.

Análisis de priorización

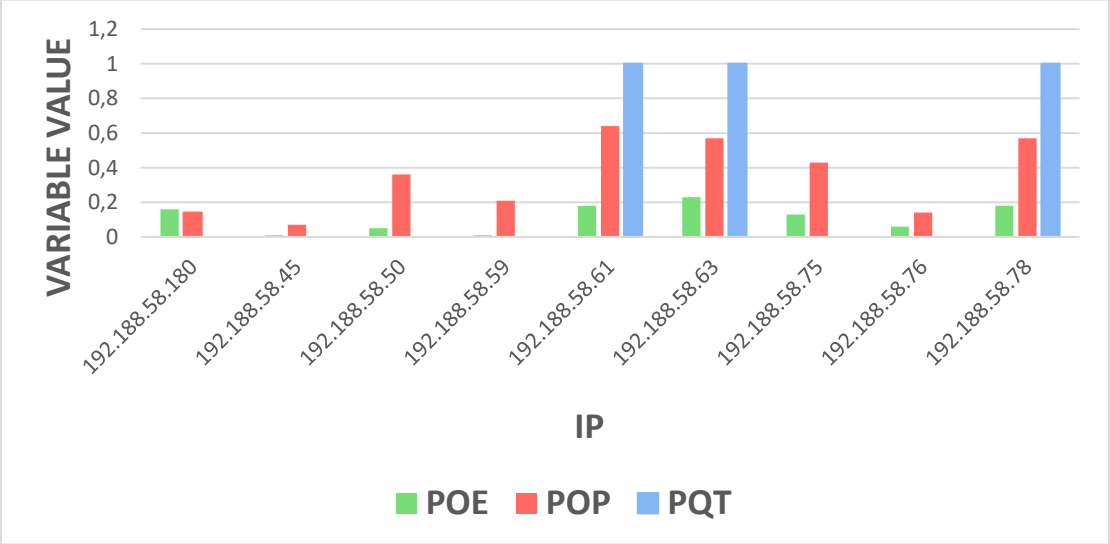
Para analizar el proceso de priorización a través del cálculo del factor de riesgo, es importante conocer los datos resultantes de las variables de entorno ya que son las que definen el orden de priorizacion. De acuerdo con el punto de vista de un atacante los vectores y formas de vulnerar un red son varias y muy ingeniosas, pero

todos los procesos de ataque comienzan con indagar la información que se tiene disponible en el amplio mundo del internet (Al-Dhaqm, 2020), (Aminanto, 2020), (Ron M. B., Fuertes, Bonilla, & Toulkeridis, 2018). El proceso previo para conocer que tan difícil resulta comprometer una organización, es definitorio para que un atacante o grupo de atacantes decidan gastar sus recursos y tiempo. Por este motivo, toda la información que las organizaciones permitan que se filtren en el internet es de vital importancia para convertirse o no en un objetivo (Goel & Nussbaum, 2021), (Sun N. Z., 2019), (CISA, 2021).

Inicialmente es importante conocer de forma general la cantidad de vulnerabilidades que la organización presenta, ya que es importante correlacionar las variables de entorno de acuerdo con las carácterísticas y cantidad de vulnerabilidades que cada IP escaneada presenta. En la Figura 14 se mostró el detalle correspondiente. Se puede observar que el máximo de vulnerabilidades contenida en un IP es de 123 y el menor es de 6, estos valores son importantes ya que nos permiten definir si la cantidad de vulnerabilidades marca una tendencia en el factor de riesgo. Farris menciona que cada IP en la red es independiente por lo cual cada valor de riesgo que tenga la IP afecta directamente al conjunto de vulnerabilidades que en ella se alojen. Bajo este enfoque nuestro modelo de priorización mantiene esta tendencia (K. A. Farris, 2018). La vulnerabilidad CVE-2017-9798 que se han presentado durante toda esta investigacion para fines de ejemplificación, se encuentra alojada en la IP-5, IP-7, IP-9 y los valores de priorización son 0.23, 0.07 y 0.22 correspondientemente.

Figura 24

Variables de IP

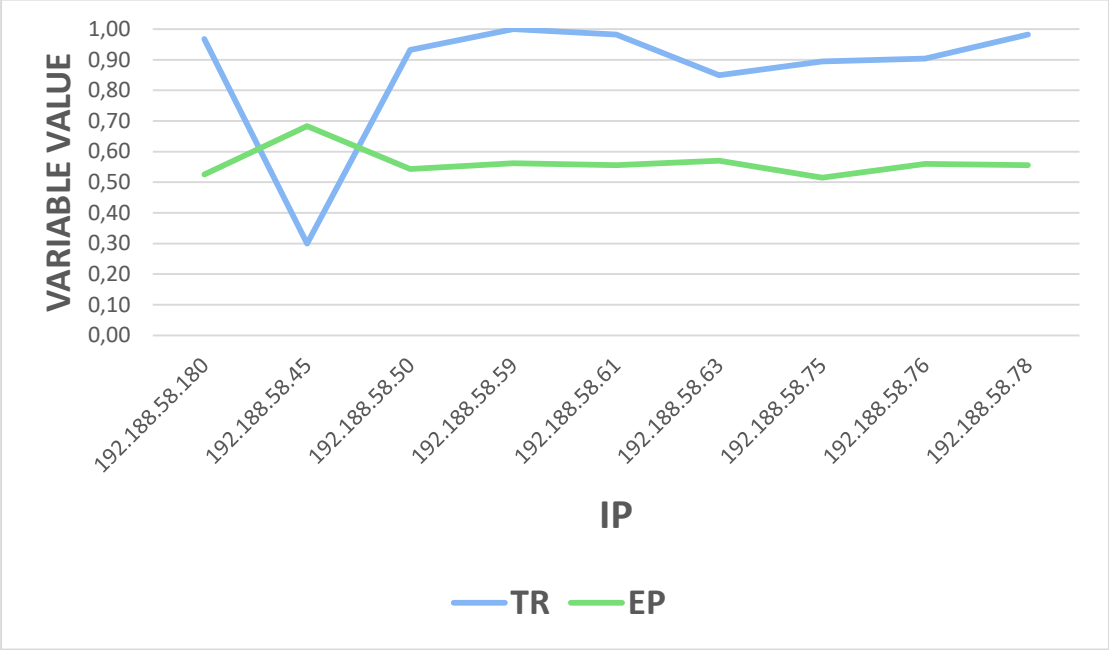


Nota. La figura representa los promedios de valores presentados por las variables de IP

En la Figura 24 se puede notar que la existencia de la variable PQT determina que el factor de riesgo aumente en una IP. Bajo este parámetro se confirma que el modelo trabaja de acuerdo con la teoría presentada en la probabilidad de las etiquetas de consulta, donde se define que Shodan cuando presenta esta variable muestra una mayor información acerca del lugar donde se aloja la vulnerabilidad, para mayor conocimiento del atacante (Lee, Shin, & Roh, 2017), (Zolotykh, 2021), por lo cual para estas IP la probabilidad de ocurrencia de esta vulnerabilidad es mayor para una prioridad más alta de atención.

Figura 25

Variables de vulnerabilidad



Nota. La figura representa los promedios de valores presentados por las variables de vulnerabilidad en cada IP.

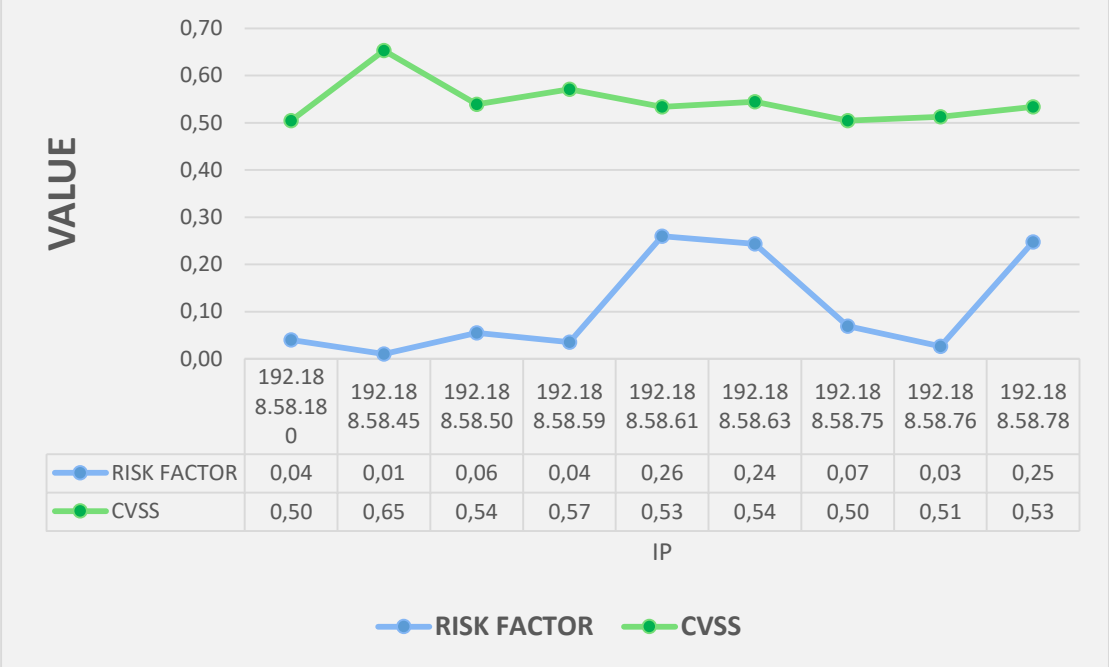
Por otro lado las variables propias que se extraen de la información de las vulnerabilidades, en la Figura 25 se puede observar la tendencia de las variables TR y EP. Se tiene que gran cantidad de las vulnerabilidades en las IP mantienen un alto porcentaje de referencias, por lo cual la probabilidad de explotación se ajusta a la baja a medida que la corrección se convierte en definitiva (Alperin, Wollaber, Ross, Trepagnier, & Leonard, 2019), (Haipeng, Liu, Liu, Park, & Subrahmanian, 2019), (Sharma & Singh, 2018).

Ademas, se observa un caso particular en la IP-2, la cuál presenta un total de 6 vulnerabilidades de las cuales ninguna supera mas de 10 referencias, y sus puntajes de CVSS en 3 de las 6 vulnerabilidades son 1 alto y 2 críticos y a pesar de tener un año pasado en el CVE-ID aun presenta un riesgo alto, lo que denota que no siempre

se van ajustar a la baja con el pasar de los años. Esto demuestra que el año de referencia en el CVE-ID es una variable de análisis considerable, ya que no tiene una tendencia fija que demuestre que un CVSS Low o Medium sobre vulnerabilidades antiguas.

Figura 26

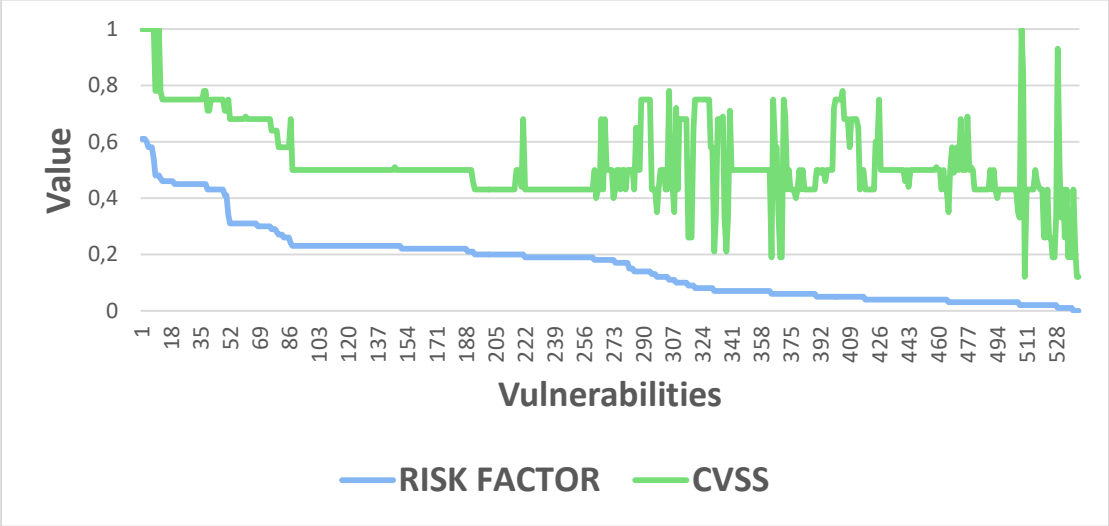
Tendencia del CVSS y factor de riesgo por IP



Nota. La figura representa la tendencia de CVSS comparada con el Factor de Riesgo por IP.

Figura 27

Tendencia del CVSS y Factor de Riesgo por vulnerabilidad



Nota. La figura representa la tendencia de CVSS comparada con el Factor de Riesgo por vulnerabilidad.

De acuerdo con esta investigación se define que las variables predominantes son las de la IP ya que como se puede observar en la Figura 26, si solo se usaría CVSS para priorizar las vulnerabilidades la IP-2 seria la primera en atenderse, sin embargo, a partir del cálculo del factor de riesgo esta IP es la de menor prioridad con 0.01. Amankwah, Dobrovoljc y Keramati mencionan que CVSS suele ser muy genérico y no atiende la especificidad necesaria para una priorizacion acertada, lo cuál se demuestra con la tendencia (Amankwan, Chen, Kwaku, Korkor, & Adutwun, 2020) (A. Dobrovoljc, 2017) (Keramati, 2016), ademas si observamos la Figura 27 es claro que el CVSS tendría una tendencia de priorización muy disfuncional de acuerdo a la organización donde se presenten.

Capítulo V

Conclusiones y Recomendaciones

Conclusiones

- Esta investigación presentó una revisión sistemática de literatura de los procesos de desarrollo enfoques, metodologías, técnicas y herramientas para los sistemas de detección de vulnerabilidades.
- Se determinó que no existe una metodología estándar o predominante para el proceso de desarrollo de VDS.
- Se identificó que el proceso metodológico de detección de vulnerabilidades debe ser iterativo para lograr resultados eficientes a través del monitoreo continuo.
- Se observó que las vulnerabilidades de software se detectan mediante técnicas de Deep Learning, Machine Learning y Fuzz. Además, las vulnerabilidades de red, IoT y aplicaciones web se apoyan en diferentes herramientas comerciales que han proporcionado resultados eficaces y eficientes.
- El proceso de desarrollo del VDS requiere de metodologías de desarrollo ágiles, evolutivas e incrementales que permitan cambios y validaciones constantes del sistema.
- La procesos de evaluación apoyan a los sistemas permitiendo la validación del rendimiento y facilita la comparación de los resultados obtenidos con herramientas comerciales.
- Se propuso un proceso de desarrollo de sistemas de vulnerabilidades que proporciona una guía para cumplir con los requisitos mínimos de escalabilidad y mejora continua.

- Se definió un modelo de priorización que se enfoca en la información que el atacante tiene disponible para comprometer una organización a través de la materialización de una vulnerabilidad.
- Los algoritmos, modelos y fórmulas matemáticas de priorización tienen problemas cuando se aplican a un entorno del mundo real, por los múltiples objetivos y que constantemente son conflictivos en las organizaciones. Los diferentes problemas tienen una enorme importancia práctica, ya que producen un conjunto de soluciones basadas en la importancia asignada a cada uno de los objetivos.
- CVSS es útil para cuantificar variables de entorno, debido a sus métricas es posible realizar comparaciones con información adicional que una organización presenta.
- Es fundamental tener un riesgo inicial con el cual se pueda abordar la criticidad de las vulnerabilidades, por lo cual CVSS no debería ser descartado de los valores de análisis.
- Cuando las vulnerabilidades no son atendidas en las organizaciones se convierten en objetivos fáciles incluso para los piratas informáticos moderadamente cualificados.
- La sobreexposición de información sobre la infraestructura tecnológica es directamente proporcional con la probabilidad de ocurrencia de un evento.
- El factor de riesgo calculado a partir de la información disponible para un atacante permite priorizar las vulnerabilidades que son visibles para cualquier usuario en internet, pero tambien permite evaluar el tipo de información y la sensibilidad de la misma.
- Shodan es un motor de busqueda potente con gran cantidad de información relevante cuando se hace uso de sus API, además se identificó que sus crawlers actualizan los datos cada dos días aproximadamente.

- El proceso de extracción, tratamiento y correlación de los datos se vuelve dinámico al lograr un monitoreo continuo, el cual permite evaluar y descubrir vulnerabilidades en tiempo real, lo cual mejora la seguridad de las infraestructuras tecnológicas.
- Se ha demostrado que las variables de entorno definen que la priorización de una vulnerabilidad debe ser evaluada por cada organización y que CVSS define por excelencia el impacto, sin embargo, la priorizacion se ajusta mejor cuando se calcula un factor de riesgo definido para cada entorno.

Recomendaciones

- Se recomienda evaluar, definir y limitar el enfoque de detección de vulnerabilidades ya que en gran medida define la especificidad del desarrollo del prototipo y hacia donde esta enfocado, ya que al ser un campo extenso se pueden obtener gran cantidad de resultados dificiles de correlacionar.
- Se recomienda documentar cada paso durante el proceso de desarrollo de los prototipos, software o herramientas, ya que esto ayuda a realizar mejoras continuas y sobretodo aprender de los errores que se puedan cometer.
- Los modelos de priorizacion son experimentales y sus resultados pueden ser validados a largo plazo, por lo cual se recomienda validar cada modelo a traves de estudios que hayan sido verificados al momento de definir variables o procesos.
- Es importante tomar en cuenta las politicas de consumo de API de plataformas como Shodan, sobre todo con CORS. Ya que al ser una plataforma abierta con un software cerrado se asegura de que las consultas no sean consecutivas si se usa una licencia básica.
- Es importante evaluar de forma manual las IP que se obtienen de Shodan, ya que muchas veces sus API no devuelven todos los valores que se puedan

investigar a traves de su plataforma por lo cual se recomienda que se tenga previamente la colección de IPs a escanear.

Referencias

A. Dobrovoljc, D. T. (2017). Predicting Exploitations of Information Systems Vulnerabilities Through Attackers. *IEEE Access, 5*, 26063–26075. doi:10.1109/ACCESS.2017.2769063

Al-Dhaqm, A. R. (2020). Towards the Development of an Integrated Incident Response Model for Database Forensic Investigation Field. *IEEE Access*, 145018–145032. doi:https://doi.org/10.1109/ACCESS.2020.3008696

Al-dhaqm, A., Abd Razak, S., & Richard, I. (2020). Towards the Development of an Integrated Incident Response Model for Database Forensic Investigation Field. *IEEE Access, 8*, 145018–145032. doi:10.1109/ACCESS.2020.3008696

Allodi, L., & Massacci, F. (octubre de 2012). A preliminary analysis of vulnerability scores for attacks in wild: the ekits and sym datasets. *Proceedings of the 2012 ACM Workshop on Building analysis datasets and gathering experience returns for security*, 17–24. doi:10.1145/2382416.2382427

Allodi, L., & Massacci, F. (agosto de 2014). Comparing Vulnerability Severity and Exploits Using Case Control Studies. *ACM Transactions on Information and System Security, 17*(1), 1:1-1:20. doi:10.1145/2630069.

Alperin, K., Wollaber, A., Ross, D., Trepagnier, P., & Leonard, L. (2019). Risk Prioritization by Leveraging Latent Vulnerability Features in a Contested Environment. *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 49–57. doi:10.1145/3338501.3357365

Alsowail, R. A.-S. (2020). Empirical Detection Techniques of Insider Threat Incidents. *IEEE Access*, 78385–78402. doi:https://doi.org/10.1109/ACCESS.2020.2989739

Alsowail, R., & Al-Shehari, T. (2020). Empirical Detection Techniques of Insider Threat Incidents. *IEEE Access, 8*, 78385–78402. doi:10.1109/ACCESS.2020.2989739

Amankwan, R., Chen, J., Kwaku, P., Korkor, B., & Adutwun, A. (Diciembre de 2020). An automated framework for evaluating open-source web scanner vulnerability severity,. *Service Oriented Computing and Applications, 14*(4), 297–307. doi:10.1007/s11761-020-00296-9.

Aminanto, M. E. (2020). Threat Alert Prioritization Using Isolation Forest and Stacked Auto Encoder With Day-Forward-Chaining Analysis. *IEEE Access*, 217977–217986. doi:https://doi.org/10.1109/ACCESS.2020.3041837

Andrade, R. O., Cordova, D., Ortiz-Garcés, I., Fuertes, W., & Cazares, M. (2020). A Comprehensive Study About Cybersecurity Incident Response Capabilities in Ecuador. *International Conference on Innovation and Research* (págs. 281-292). Springer, Cham.

Antunes, N., & Vieira, M. (2015). Assessing and Comparing Vulnerability Detection Tools for Web Services: Benchmarking Approach and Examples. *IEEE TRANSACTIONS ON SERVICES COMPUTING, 8*(2), 269-284.

BA, K. &. (2007). Guidelines for performing Systematic Literature Reviews in Software Engineering. 2.

C.B., Ş., Ö.B., D., & Abualigah, L. (2021). Prediction of software vulnerability based deep symbiotic genetic algorithms: Phenotyping of dominant-features. *ELSEVIER Applied Intelligence*.

Cao, S., Sun, X., Bo, L., Wei, Y., & Li, B. (2021). BGNN4VD: Constructing Bidirectional Graph Neural-Network for Vulnerability Detection. *ELSEVIER Information and Software Technology*.

Cigoj, P. &. (2019). *An Intelligent and Automated WCMS Vulnerability-Discovery Tool: The Current State of the Web*. Obtenido de https://doi.org/10.1109/ACCESS.2019.2957573

CISA. (23 de octubre de 2021). *Remediate Vulnerabilities for Internet-Accessible Systems*. Obtenido de https://www.cisa.gov/publication/remediate-vulnerabilities-internet-accessible-systems

Cisco. (18 de Enero de 2021). *Cisco*. Obtenido de Cisco: https://www.cisco.com/c/es_mx/products/security/what-is-cybersecurity.html

CVE. (11 de 12 de 2020). *CVE*. Obtenido de CVE: https://cve.mitre.org/about/cve_and_nvd_relationship.html

CVE. (07 de septiembre de 2021). Obtenido de https://cve.mitre.org/

CVSS. (2015). Common Vulnerability Scoring System version 3.1: Specification Document. *Incident Response and Security Teams*. Obtenido de https://www.first.org/cvss/specification-document

CVSS. (09 de septiembre de 2019). Common Vulnerability Scoring System versión 3.1: Documento de especificación. *First Improving Segurity Together*. Obtenido de https://www.first.org/cvss/v3.1/specification-document

Deb, R., & Roy, S. (marzo de 2020). Dynamic vulnerability assessments of software-defined networks. *Innovaciones en Ingeniería de Sistemas y Software, 16*(1), 45–51. doi:10.1007/s11334-019-00337-3

Dondo , M. G. (2008). A Vulnerability Prioritization System Using A Fuzzy Risk Analysis Approach. *Springer*, 525–540. doi:10.1007/978-0-387-09699-5 34

Dondo, M. G. (2008). A Vulnerability Prioritization System Using A Fuzzy Risk Analysis Approach. *Proceedings of The Ifip Tc 11 23rd International Information Security Conference*, 525–540. doi: 10.1007/978-0-387-09699-5 34

ENISA. (2006). *A STEP-BY-STEP APPROACH ON HOW TO SET UP A CSIRT.*

ENISA. (n.d. de n.d. de 2021). *ENISA*. Obtenido de ENISA: https://www.enisa.europa.eu/topics/csirt-cert-services/quality-management

Erza Aminanto, M., Ban, T., Isawa, R., Takahashi, T., & Inoue, D. (2020). Threat Alert Prioritization Using Isolation Forest and Stacked Auto Encoder With Day-Forward-Chaining Analysis. *IEEE Access, 8*, 217977–217986. doi:10.1109/ACCESS.2020.3041837

ESET. (23 de MAYO de 2018). *ESET*. Obtenido de ESET: https://www.eset.com/mx/acerca-de-eset/sala-de-prensa/comunicados-de-prensa/press-releases/67-de-las-instituciones-educativas-aseguro-haber-sufrido-al-menos-un-incidente-de-seguridad-1/

ESET. (n.d. de n.d. de 2020). *Welivesecurity*. Obtenido de Welivesecurity: https://www.welivesecurity.com/wp-content/uploads/2020/08/ESET-Security-Report-LATAM_2020.pdf

Feutrill, A., Ranathunga, D., Roughan, M., & Yarom, Y. (noviembre de 2018). The Effect of Common Vulnerability Scoring System Metrics on Vulnerability Exploit Delay. *Sixth International Symposium on Computing and Networking (CANDAR)*, 1–10. doi:10.1109/CANDAR.2018.00009

FIRST. (n.d. de n.d. de 2021). *FIRST*. Obtenido de FIRST: https://www.first.org/

Frei, S. (octubre de 2009). Security Econometrics - The Dynamics of (In)Security. Obtenido de https://www.researchgate.net/publication/230757822_Security_Econometrics_-_The_Dynamics_of_InSecurity

G. Spanos, L. A. (2017). Assessment of Vulnerability Severity using Text Mining . *Proceedings of the 21st Pan-Hellenic Conference on Informatics*, 1–6. doi:10.1145/3139367.3139390

Galindo, P., Nespoli, P., Gomez, F., & Martinez, G. (2020). The Not Yet Exploited Goldmine of OSINT: Opportunities, Open Challenges and Future Trends. *IEEE Access, 8*, 10282–10304. doi:10.1109/ACCESS.2020.2965257

Ghaffarian, S. M., & Shahriari, H. R. (2017). Software Vulnerability Analysis and Discovery Using Machine-Learning and Data-Mining Techniques: A Survey. *ACM Computing Surveys, 50*(4).

Goel, S. &. (2021). Attribution Across Cyber Attack Types: Network Intrusions and Information Operations. *IEEE Open Journal of the Communications Society,*, 1082–1093. doi:https://doi.org/10.1109/OJCOMS.2021.3074591

Goel, S., & Nussbaum, B. (2021). Attribution Across Cyber Attack Types: Network Intrusions and Information Operations. *IEEE Open Journal of the Communications Society, 2*, 1082-1093. doi:10.1109/OJCOMS.2021.3074591

H., W., G., Y., Z., T., S.H., T., S., H., D., F., . . . Z, W. (2021). Combining Graph-Based Learning with Automated Data Collection for Code Vulnerability Detection. *IEEE Transactions on Information Forensics and Security, 16*(9293321), 1943-1958.

Hackmageddon. (n.d. de n.d. de 2020). *Hackmageddon*. Obtenido de Hackmageddon: https://www.hackmageddon.com/2021/01/11/december-2020-cyber-attacks-statistics/

Haipeng, C., Liu, J., Liu, R., Park, N., & Subrahmanian, V. (noviembre de 2019). VASE: A Twitter-Based Vulnerability Analysis and Score Engine. *IEEE International Conference on Data Mining (ICDM)*, 976–981. doi:10.1109/ICDM.2019.00110.

Han, L., Zhou, M., Qian, Y., Fu, C., & Zou, D. (2019). An Optimized Static Propositional Function Model to Detect Software Vulnerability. *IEEE Access, 7*, 143499-143510.

Heide, M. v. (2017). *Establishing a CSIRT.* Thailand: Thailand Computer Emergency Response Team.

Hellwig, O., Quirchmayr, G., Huber, E., Goluch, G., Vock, F., & Pospisil, B. (2016). Major Challenges in Structuring and Institutionalizing CERT-Communication. *IEEE*.

Hu, H., Zhang, H., & Yang, Y. (2018). Security risk situation quantification method based on threat prediction for multimedia communication network. *Multimedia Tools and Applications volume, 14*, 21693–21723.

IBM Corporation. (2020). *IBM Security. New York: IBM Corporation. IBM Security*. Obtenido de https://www.ibm.com/security/digital-assets/cost-data-breach-report/#/es-mx

INCIBE. (20 de 03 de 2017). *INCIBE*. Obtenido de INCIBE: https://www.incibe.es/protege-tu-empresa/blog/amenaza-vs-vulnerabilidad-sabes-se-diferencian

INCIBE-CERT. (21 de julio de 2015). Metricas de evaluaci ´ on de vulnerabilidades: CVSS 3.0," ´. *INCIBE-CERT*. Recuperado el 13 de noviembre de 2021, de https://www.incibe-cert.es/blog/cvss3-0

Jeon, S., & Kang, H. (julio de 2021). AutoVAS: An automated vulnerability analysis system with a deep learning approach. *Computers & Security, 106*, 102308. doi:10.1016/j.cose.2021.102308.

Jeon, S., & Kim, H. K. (2021). AutoVAS: An automated vulnerability analysis system with a deep learning approach. *ELSEVIER Computers & Security*.

K. A. Farris, A. S. (junio de 2018). VULCON: A System for Vulnerability Prioritization, Mitigation, and Management. *ACM Transactions on Privacy and Security, 21*(4), 16:1-16:28. doi: 10.1145/3196884

K. Alperin, A. W. (noviembre de 2019). Risk Prioritization by Leveraging Latent Vulnerability Features in a Contested Environment. *Proceedings of the 12th ACM Workshop on Artificial Intelligence and Security*, 49–57. doi:10.1145/3338501.3357365

Keramati, M. (septiembre de 2016). New Vulnerability Scoring System for dynamic security evaluation. *International Symposium on Telecommunications (IST)*, 746–751. doi: 10.1109/ISTEL2016.7881922

Kitchenham, B., & Charters, S. (2007). Guidelines for performing Systematic Literature Reviews in Software Engineering. *EBSE 2007-001. Keele University and Durham University Joint Report*.

Lee, S., Shin, S.-H., & Roh, B.-h. (junio de 2017). Abnormal Behavior-Based Detection of Shodan and Censys-Like Scanning. *Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, 1048–1052. doi:10.1109/ICUFN.2017.7993960

Li, Z. Z. (2019). *A Comparative Study of Deep Learning-Based Vulnerability Detection System. IEEE Access*. Obtenido de https://doi.org/10.1109/ACCESS.2019.2930578

Liu, M., & Wang, B. (2018). A Web Second-Order Vulnerabilities Detection Method. *IEEE Access, 6*, 70983-70989.

Liu, S., Lin, G., Han, Q.-L., Wen, S., Zhang, J., & Xiang, Y. (2019). DeepBalance: Deep-Learning and Fuzzy Oversampling for Vulnerability Detection. *IEEE Transactions on Fuzzy Systems*.

Mena, F. X., Díaz, W. M., Jaramillo, C. E., Estévez, E. P., Barzallo, P. F., & Silva, C. J. (2018). Application of business intelligence for analyzing vulnerabilities to increase the security level in an academic CSIRT. *Revista Facultad de Ingeniería*, 21-29.

Meng, G., Liu, Y., Zhang, J., Pokluda, A., & Boutaba, R. (2015). Collaborative Security: A Survey and Taxonomy. *ACM Computing Surveys, 48*(1).

NIST. (18 de 9 de 2012). *NIST.* Obtenido de NIST: https://csrc.nist.gov/csrc/media/publications/sp/800-30/rev-1/final/documents/sp800-30-rev1-ipd.pdf

NIST. (18 de Enero de 2021). *CENTRO DE RECURSOS DE SEGURIDAD INFORMÁTICA CSRC*. Obtenido de CENTRO DE RECURSOS DE SEGURIDAD INFORMÁTICA CSRC: https://csrc.nist.gov/glossary/term/cybersecurity#:~:text=1%20under%20Cybersecurity%20CNSSI%204009,detecting%2C%20and%20responding%20to%20attacks.

OEA, B. (n.d. de n.d. de 2020). *Inter - American Development Bank*. Obtenido de Inter - American Development Bank: https://publications.iadb.org/publications/spanish/document/Reporte-Ciberseguridad-2020-riesgos-avances-y-el-camino-a-seguir-en-America-Latina-y-el-Caribe.pdf

OEA, B. (n.d. de n.d. de n.d.). *Observatorio de la Ciberseguridad en América Latina y El Caribe*. Obtenido de Observatorio de la Ciberseguridad en América Latina y El Caribe: https://observatoriociberseguridad.org/

Qasemen, A., Shirani, P., Debbabi, M., Wang, L., Lebel, B., & Agba, B. L. (2021). Automatic Vulnerability Detection in Embedded Devices and Firmware: Survey and Layered Taxonomies. *ACM Computing Surveys, 54*(2), 25.

Qiang, W. L. (2017). *Patchrelated Vulnerability Detection Based on Symbolic Execution*. Obtenido de https://doi.org/10.1109/ACCESS.2017.2676161

Qin, J., Zhang, H., Guo, J., Wang, S., Wen, Q., & Shi, Y. (2016). Vulnerability Detection on Android Apps–Inspired by Case Study on Vulnerability Related with Web Functions. *IEEE Access, 4*.

R. Amankwah, J. C. (Diciembre de 2020). An automated framework for evaluating open-source web scanner vulnerability severity,. *Service Oriented Computing and Applications, 14*(4), 297–307. doi:10.1007/s11761-020-00296-9.

R. Sharma, R. S. (2019). Software Vulnerability Prioritization: A Comparative Study Using TOPSIS and VIKOR Techniques. *System Performance and Management Analytics*, 405–418. doi:10.1007/978-981-10-7323-6 32

R. Sharma, R. S. (2019). Software Vulnerability Prioritization: A Comparative Study Using TOPSIS and VIKOR Techniques. *System Performance and Management Analytics*, 405–418. doi: 10.1007/978-981-10-7323-6 32

Rajasooriya, S. M., Tsokos, C. P., & Kaluarachchi, P. K. (abril de 2017). Cyber Security: Nonlinear Stochastic Models for Predicting the Exploitability. *Journal of Information Security, 8*(2). doi:10.4236/jis.2017.82009

Ren, Y. D. (2019). A Dynamic Taint Analysis Framework Based on Entity Equipment. *Applied Intelligence*, 186308–186318. Obtenido de https://ieeexplore.ieee.org/document/8937477/

Ron, M. B., Fuertes, W., Bonilla, M., & Toulkeridis, T. (junio de 2018). 13th Cybercrime in Ecuador, an exploration, which allows to define national cybersecurity policies. *Iberian Conference on Information Systems and Technologies (CISTI)*, 1–7. doi:10.23919/CISTI.2018.8399357

Ron, M., Fuertes, W., Bonilla, M., Toulkeridis, T., & Diaz, J. (2018). Cybercrime in Ecuador, an exploration, which allows to define national cybersecurity policies. *IEEE*, 1-7.

S¸ahin, C. B. (2021). Prediction of software vulnerability based deep symbiotic genetic algorithms: Phenotyping of dominant-features. *Applied Intelligence*. Obtenido de https://doi.org/10.1007/s10489-021-02324-3

Securelist. (03 de Septiembre de 2020). *Securelist by Kaspersky*. Obtenido de Securelist by Kaspersky: https://securelist.lat/it-threat-evolution-q2-2020-pc-statistics/90949/

Sharma , R., Sibal, R., & Sabharwal, S. (julio de 2020). Software vulnerability prioritization using vulnerability description. *International Journal of System Assurance Engineering and Management, 12*. doi:10.1007/s13198-020-01021-7

Sharma, R., & Singh, R. (2018). An Improved Scoring System for Software Vulnerability Prioritization. (U. K. P. K. Kapur, Ed.) *Quality, IT and Business Operations: Modeling and Optimization*, 33–43. doi:10.1007/978-981-10-5577-5 3

Shodan. (septiembre de 07 de 2021). Obtenido de https://www.shodan.io

Shukla, A., Katt, B., & Obiora, N. (deciembre de 2019). Vulnerability Discovery Modelling With Vulnerability Severity. *IEEE Conference on Information and Communication Technology*, 1–6. doi:10.1109/CICT48419.2019.9066187

Sõnmez, F. Ö., & Kilic, B. G. (2021). Holistic Web Application Security Visualization for Multi-Project and Multi-Phase Dynamic Application Security Test Results. *IEEE Access, 9*, 25858-25884.

SOPHOS. (18 de Enero de 2020). *INFORME DE AMENAZAS 2021 DE SHOPHOS.* Obtenido de SHOPHOS: https://www.sophos.com/es-es/medialibrary/pdfs/technical-papers/sophos-2021-threat-report.pdf

Sun, N. Z. (2019). Data Driven Cybersecurity Incident Prediction: A Survey. *IEEE Communications Sur veys Tutorials*, 1744–1772. doi:https://doi.org/10.1109/COMST.2018.2885561

Sun, N., Zhang, J., Rimba, P., Gao, S., Yu Zang, L., & Xiang, Y. (2019). Data-Driven Cybersecurity Incident Prediction: A Survey. *IEEE Communications Surveys Tutorials, 21*(2), 1744–1772. doi:10.1109/COMST.2018.2885561

Vielberth, M. B. (2020). Security Operations Center: A Systematic Study and Open Challenges. *IEEE Access*, 227756–227779. doi:https://doi.org/10.1109/ACCESS.2020.3045514

Vielberth, M., Bohm, F., Fichtinger, I., & Günther, P. (2020). Security Operations Center: A Systematic Study and Open Challenges. *IEEE Access, 8*, 227756–227779. doi:10.1109/ACCESS.2020.3045514

Wang, H. Y. (2021). Combining Graph-Based Learning With Automated Data Collection for Code Vulnerability Detection IEEE Transactions on Information. *Forensics and Security*, 1943–1958. doi:https://doi.org/10.1109/TIFS.2020.3044773

Wang, M. L. (2018). A Web Second-Order Vulnerabilities Detection Method. *IEEE Access, 6*, 70983–70988. doi:10.1109/ACCESS.2018.2881070

Yi, M., Xu, X., & Xu, L. (2017). An Intelligent Communication Warning Vulnerability Detection Algorithm Based on IoT Technology. *IEEE Access*.

Yu, L. L. (2021). BEDetector: A Two Channel Encoding Method to Detect Vulnerabilities Based on Binary Similarity. *IEEE Access*, 51631–51645. doi:https://doi.org/10.1109/ACCESS.2021.3064687

Yu, M., Zhuge, J., Cao, M., Shi, Z., & Jiang, L. (2020). A Survey of Security Vulnerability Analysis, Discovery, Detection, and Mitigation on IoT Devices. *Future Internet MDPI*(12020027), 12-27.

Zagane, M., Abdi, M. K., & Alenezi, M. (2020). Deep Learning for Software Vulnerabilities Detection Using Code Metrics. *IEEE Access, 8*, 74562-74571.

Zeng, P., Lin, G., Pan, L., Tai, Y., & Zhang, J. (2020). Software Vulnerability Analysis and Discovery Using Deep Learning Techniques: A Survey. *IEEE Access, 8*, 197158-197172.

Zhang, H., & Sakurai, K. (2021). A Survey of Software Clone Detection From Security Perspective. *IEEE Access, 9*, 48157-48173.

Zheng, W., Gao, J., Wub, X., Liu, F., Xun, Y., & Liu, G. (2020). The impact factors on the performance of machine learning-based vulnerability detection: A comparative study. *ELSEVIER The Journal of Systems & Software*.

Zhu, K. L. (2020). Scalable Static Detection of Use After-Free Vulnerabilities in Binary Code. *IEEE Access*, 78713–78725. doi:https://doi.org/10.1109/ACCESS.2020.2990197

Zolotykh, M. (mayo de 2021). Study of Crawlers of Search Engine ‘Shodan.io. *Ural Symposium on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT)*, 0419–0422. doi: 10.1109/USBEREIT51232.2021.9455018