



Programa de Impulso a la Industria de la Ciberseguridad Nacional

#INCIBEemprende



INCIBE Emprende desarrolla iniciativas de **ideación, incubación y aceleración** tanto para la promoción del emprendimiento en ciberseguridad, como de la ciberseguridad en el emprendimiento de base tecnológica, con el fin de **Impulsar la Industria Nacional de Ciberseguridad**.

- ◆ **Acompañamos** a emprendedores y start-ups españolas de **ciberseguridad, así como de base tecnológica** para la integración de la ciberseguridad en su proyecto.
- ◆ **Impulsamos el desarrollo de proyectos** independientemente de su grado de madurez: fomento de ideas - incubación de proyectos - aceleración de start-ups e internacionalización.
- ◆ Impulsamos la **innovación** y promocionamos la atracción de **inversión** a la misma.

Plan de Recuperación, Transformación y Resiliencia (PRTR) a través del Componente 15. Inversión 7 Ciberseguridad: Fortalecimiento de las capacidades de ciudadanos, PYMES y profesionales e impulso del sector.

Start-ups vs emprendimiento tradicional



- ◆ Empresas digitales/base tecnológica
- ◆ Explotación de nuevos modelos de negocio
- ◆ Modelo de negocio escalable: crecimiento exponencial en ventas y crecimiento lineal en costes
- ◆ Ventaja competitiva basada en la innovación
- ◆ Alto riesgo debido a su alto contenido innovador
- ◆ Necesidad de elevado volumen de financiación y dificultad de acceso a la misma en fases iniciales
- ◆ Fuentes de financiación FFF (family, fools and Friends), capital riesgo, business angels. La garantía la supone el equipo.
- ◆ Dependencia en captación y retención de empleados altamente cualificados.

VS



- ◆ Explotación de modelos de negocio convencionales
- ◆ Crecimiento estable, modelo de negocio sostenible
- ◆ Fuente de financiación tradicional: préstamos y recursos propios. Garantía real y/o personal.
- ◆ Centrada en mantener el negocio y/o generar un crecimiento % estable.

Sector de la ciberseguridad

SECTOR EN AUGE

122.284

Trabajadores empleados (2021)

24.119

Brecha de talento (2021)

83.000

Brecha de talento (2024)



Participación mundial en la industria de ciberseguridad

31% 69%



Participación estudiantes grados STEM

24% 76%



Participación estudiantes cursos ciberseguridad

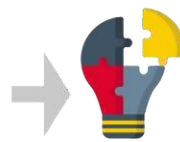
18% 82%

VALOR ECOSISTEMA*



1.500M

Año 2020



8,12%

Crecimiento anual



2.000M

Previsión 2024

*Fuente: Security Spending Guide IDC 2021

Sector de la ciberseguridad



Ciberseguridad

La importancia de la ciberseguridad es **transversal** a todos los sectores de la economía



Digitalización sectores

La digitalización de todos los sectores está favoreciendo un **crecimiento global del mercado de la Ciberseguridad**



El sector que más peso aporta a la ciberseguridad es el de **distribución y servicios**, seguido del **financiero**

*Fuente: Security Spending Guide IDC 2021

Programa de apoyo al Emprendimiento en ciberseguridad



incibe emprende 2023-2026

Captación / Ideación

Charlas
Talleres
Eventos

Incubación

Formación
Mentorización
Asesoramiento legal
Asesor fiscal
Visita al ecosistema regional
Demo Day

Aceleración

Formación
Mentorización
Asesoramiento legal
Asesor fiscal
Visita al ecosistema regional
Demo Day
Aceleración Express

Captación / Ideación

Desarrollo y Despliegue seguro de aplicaciones

- Antonio Reche – Solutions Architect Veracode
- Cómo evitar vulnerabilidades en hacer que el código de tus desarrollos no sea vulnerable. Los hagas tú o te los hagan.



Charla



Introduce aquí el contenido de la charla correspondiente



VERACODE

Desarrollo y Despliegue seguro de aplicaciones : El impacto de la IA en la Ciberseguridad

Antonio Reche –Veracode

Las dos caras de la IA

Seguridad

Hackers



La forma de
crear software
ha cambiado
para siempre



La adopción de
devops



La adopción de
cloud infrastructure



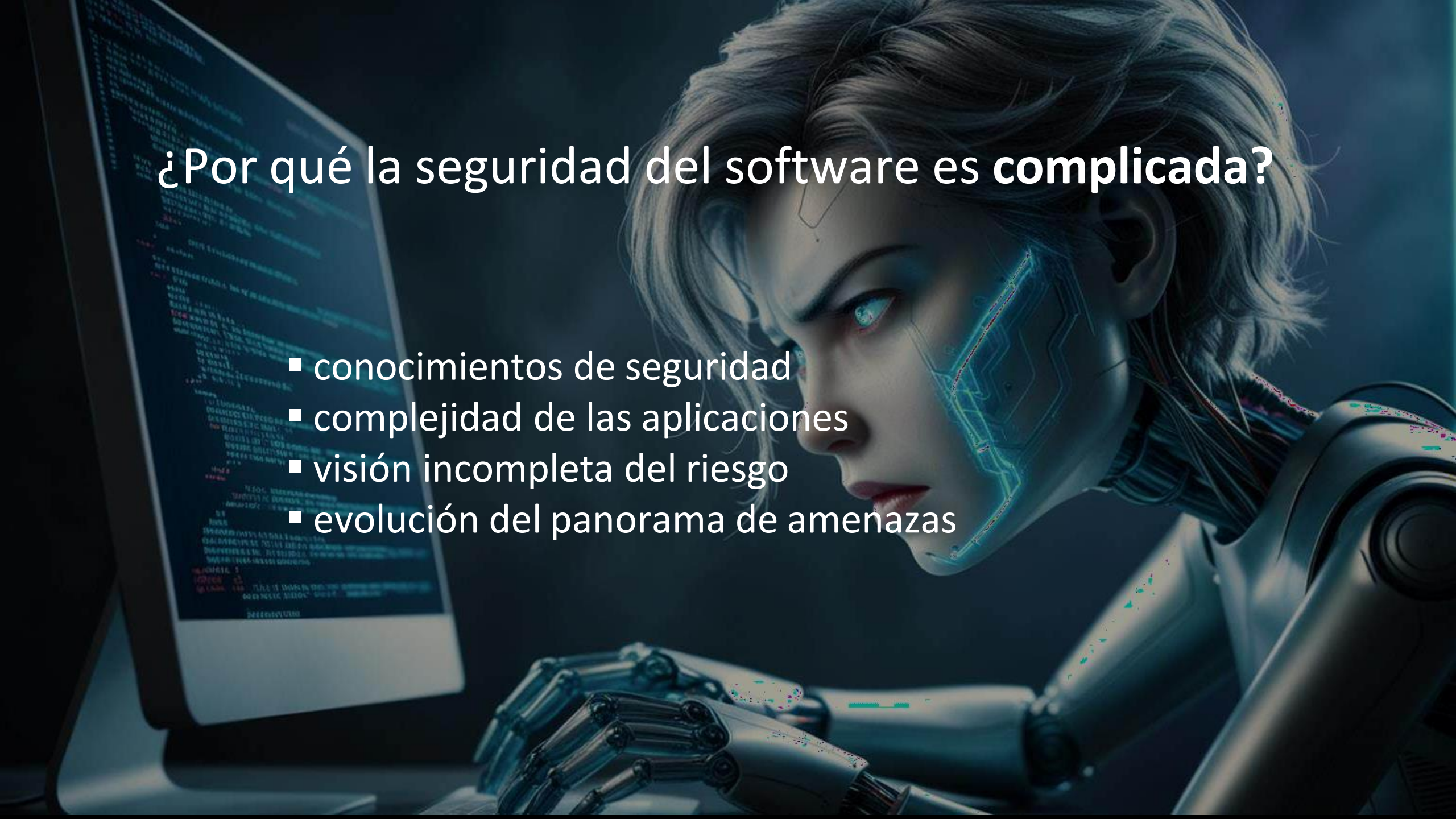
La adopción de
secure by design



¿Por qué la seguridad
del software es
complicada?

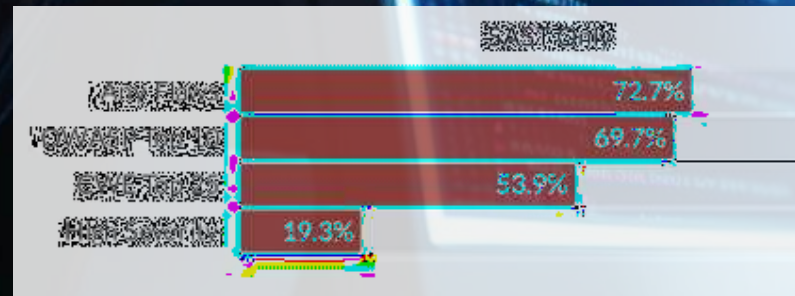
¿Por qué la seguridad del software es complicada?

- conocimientos de seguridad
- complejidad de las aplicaciones
- visión incompleta del riesgo
- evolución del panorama de amenazas



Mi código es seguro.
Mis aplicaciones son seguras.
No voy a sufrir una incidencia.

VERACODE SOSS REPORT 2024



Code Injection (SQL Injection, NoSQL Injection, Command Injection, etc.) Cross-Site Scripting (XSS) Cross-Site Request Forgery (CSRF) Inyección de Código de Deserialización Insegura Exposición de Datos Sensibles Configuración de Seguridad Incorrecta Fallos de Autenticación y Gestión de Sesiones Control de Acceso Insuficiente Inseguridad en APIs Uso de Componentes con Vulnerabilidades Conocidas Logging y Monitoreo Insuficiente Seguridad en el Transporte Insuficiente Inyección de Dependencias Exposición de Servicios Internos ... y un largo etc ...

MOVEit Transfer Vuln (CVE-2023-34362) Spring4Shell (CVE-2022-22965) Follina (CVE-2022-30190) Log4Shell (CVE-2021-44228) SolarWinds Orion (CVE-2020-10148) BlueKeep (CVE-2019-0708) jQuery (CVE-2020-11022 y CVE-2020-11023) Jackson Databind (Múltiples CVEs) pache Struts (CVE-2017-5638) Heartbleed (CVE-2014-0160)... etc...


las organizaciones se ahogan en las deudas de seguridad



* We are defining all flaws that remain unremediated for over one year, regardless of severity, as security debt.
**Critical debt: High-severity flaws that remain unremediated for over one year.



pocos equipos
corrigen los fallos de seguridad lo
suficientemente rápido como para
reducir el riesgo de seguridad a un
ritmo significativo



una buena seguridad
del software **protege**
lo importante

buena seguridad del software



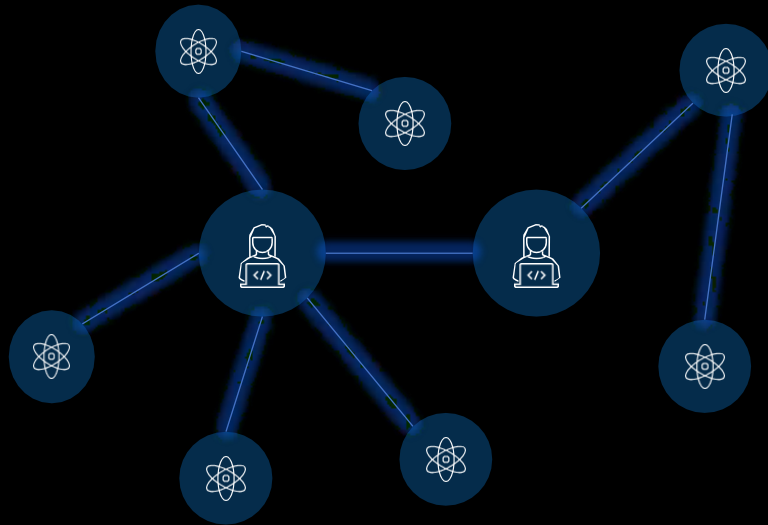
proteger lo importante



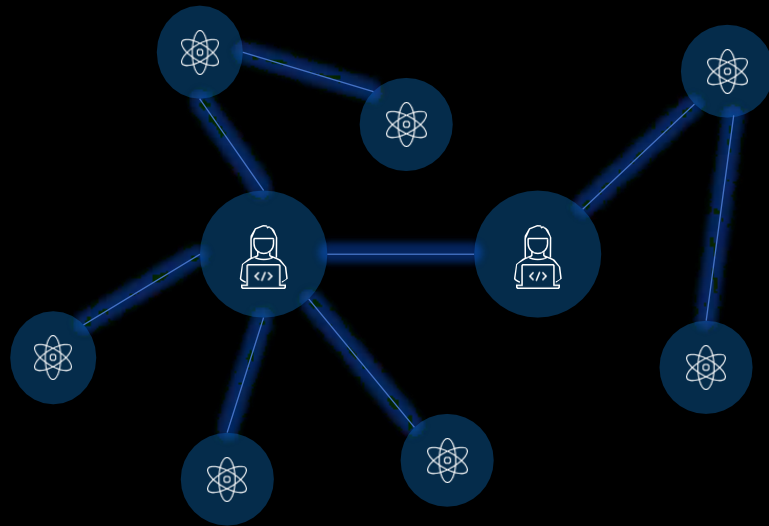
cómo creamos software

ha cambiado para siempre

las aplicaciones se
ensamblan, **no sólo se
escriben**



El 80% del código de las aplicaciones modernas es **código abierto**



cada paquete de OSS
trae consigo otros 77
de forma transitoria

para ir más rápido los
desarrolladores
aprovechan

Open source

Repos de código

Large language models

Generadores de código

A futuristic robot with a glowing blue eye and a circular light on its head is sitting on the floor. Behind it is a large, curved screen displaying lines of code in a dark font. The scene is lit with blue light, creating a high-tech atmosphere.

**Entrenar
modelos con
código abierto**

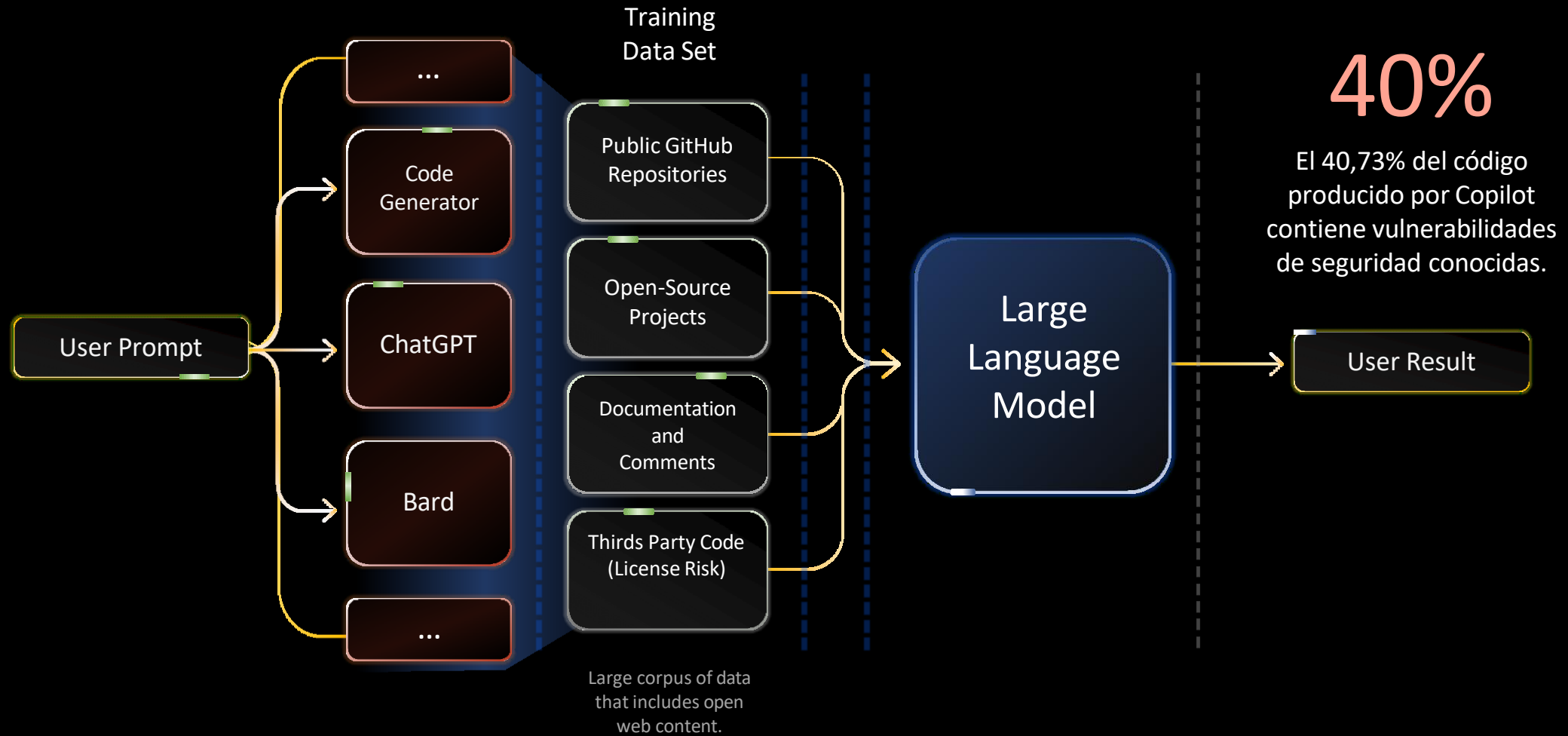
vs

**Eliminar cualquier
vulnerabilidad
antes de entrenar
el modelo**

Una encuesta de Gartner revela que el **45% de los ejecutivos** afirma que **ChatGPT** ha provocado un **aumento de la inversión en IA**

El 70% de las organizaciones se encuentra actualmente en modo de exploración con IA generativa

Large Language Models



Implicaciones de seguridad de los LLMs

Estudio de la Universidad de Cornell sobre generadores de código de IA

Estudio de la Universidad de Stanford sobre generadores de código de IA

Estudio de la Universidad de Nueva York sobre la precisión de ChatGPT en GitHub Copilot

Universidad de Purdue sobre la precisión de ChatGPT

35%

De los 435 fragmentos de código generados por Copilot, el 35% contienen fallos de seguridad, en 6 lenguajes de programación.

52%

El 52% de las respuestas de ChatGPT contienen imprecisiones y el 77% son absurdas cuando responden a preguntas de ingeniería de software.

39%

El 39,34% de los usuarios sigue prefiriendo las respuestas de ChatGPT por su exhaustividad y su estilo articulado.

75%

El 75% de los desarrolladores han afirmado que GitHub CoPilot les hace codificar más rápido.

arXiv:2310.02059v1 [cs.SE] 3 Oct 2023

Security Weaknesses of Copilot Generated Code in GitHub

Vijia Fu, Peng Liang, Anujed Tahir
School of Computer Science, School of Mathematical and Computational Science, School of Mathematical and Computational Science
Wuhan University, Wuhan, China, Wuhan, China, Fribourg, Switzerland
vijiaf@whu.edu.cn, liangp@whu.edu.cn, anujed@fhnw.ch

Zengyang Li, Mojaba Shahin, Jianlin Yu
School of Computer Science, School of Computer Science, School of Computer Science
Central China Normal University, Wuhan University, Wuhan University
Wuhan, China, Wuhan, China, Wuhan, China
zyengyang@ccnu.edu.cn, mohajaba@whu.edu.cn, jianlinyu@whu.edu.cn

ABSTRACT
Machine code generation tools like AI models, particularly Large Language Models (LLMs), can generate functional and complete code. While such tools can be promising for developers and industry developers, using these tools without comprehensive security checks risks introducing vulnerabilities into the code. This study conducted an empirical study to evaluate the security of code generated by Copilot. The results showed that Copilot generated code contains security weaknesses, particularly in the areas of authentication, authorization, and data handling. The study also identified several common security weaknesses in the generated code, such as hard-coded credentials, weak password requirements, and lack of input validation. The study also identified several common security weaknesses in the generated code, such as hard-coded credentials, weak password requirements, and lack of input validation. The study also identified several common security weaknesses in the generated code, such as hard-coded credentials, weak password requirements, and lack of input validation.

CCS CONCEPTS
Security and encryption → Software security technologies → Security and privacy → Software security engineering

KEYWORDS
Code Generation, Security Weaknesses, LLMs, GitHub Copilot

ACM Reference Format:
Vijia Fu, Peng Liang, Anujed Tahir, Zengyang Li, Mojaba Shahin, Jianlin Yu. 2023. Security Weaknesses of Copilot Generated Code in GitHub. In Proceedings of the ACM Conference on Computer Security (CCS ’23), New York, NY, USA, 10 pages. https://doi.org/10.1145/3622383

1 INTRODUCTION
Code generation tools like AI models, particularly Large Language Models (LLMs), can generate functional and complete code. While such tools can be promising for developers and industry developers, using these tools without comprehensive security checks risks introducing vulnerabilities into the code. This study conducted an empirical study to evaluate the security of code generated by Copilot. The results showed that Copilot generated code contains security weaknesses, particularly in the areas of authentication, authorization, and data handling. The study also identified several common security weaknesses in the generated code, such as hard-coded credentials, weak password requirements, and lack of input validation. The study also identified several common security weaknesses in the generated code, such as hard-coded credentials, weak password requirements, and lack of input validation.

arXiv:2211.03622v3 [cs.CR] 18 Dec 2023

Do Users Write More Insecure Code with AI Assistants?

Neil Perry*, Megha Srivastava*, Deepak Kumar, Dan Roesch
Stanford University, Stanford University, Stanford University, Stanford University

ABSTRACT
AI code assistants have emerged as powerful tools that can aid in the software development lifecycle and can improve developer productivity. Unfortunately, such assistants have also been found to produce insecure code in lab environments, raising significant concerns about their usage in practice. In this paper, we conduct a user study to evaluate how users interact with AI code assistants to write a variety of security-related code. Overall, we find that participants who had access to an AI assistant wrote significantly less secure code than those without access to an assistant. Participants who had access to an AI assistant were also more likely to believe that their AI-generated code was more secure than their own code. We also find that participants who had access to an AI assistant were more likely to believe that their AI-generated code was more secure than their own code.

CCS CONCEPTS
Security and encryption → Software security technologies → Security and privacy → Software security engineering

KEYWORDS
Code Generation, Security Weaknesses, LLMs, GitHub Copilot

ACM Reference Format:
Neil Perry, Megha Srivastava, Deepak Kumar, Dan Roesch. 2023. Do Users Write More Insecure Code with AI Assistants? In Proceedings of the ACM Conference on Computer Security (CCS ’23), New York, NY, USA, 10 pages. https://doi.org/10.1145/3622383

1 INTRODUCTION
AI code assistants, like GitHub Copilot, have emerged as powerful tools that can aid in the software development lifecycle and can improve developer productivity. Unfortunately, such assistants have also been found to produce insecure code in lab environments, raising significant concerns about their usage in practice. In this paper, we conduct a user study to evaluate how users interact with AI code assistants to write a variety of security-related code. Overall, we find that participants who had access to an AI assistant wrote significantly less secure code than those without access to an assistant. Participants who had access to an AI assistant were also more likely to believe that their AI-generated code was more secure than their own code. We also find that participants who had access to an AI assistant were more likely to believe that their AI-generated code was more secure than their own code.

arXiv:2108.09293v3 [cs.CR] 16 Dec 2021

Asleep at the Keyboard? Assessing the Security of GitHub Copilot’s Code Contributions

Hammadreyya Purohit, Balogh Ahmad, Benjamin Tan, Brandon Dolan-Gold, Ramesh Korthi
Department of ECE, Department of ECE, Department of ECE, Department of ECE, Department of ECE
Brookline, NY, USA, Brookline, NY, USA, Calgary, Alberta, CA, New York University, New York University, New York University, Brookline, NY, USA, Brookline, NY, USA

ABSTRACT
There is a burgeoning interest in designing AI-based assistants to assist humans in designing computer systems. However, such assistants generate code that is not always secure. This paper assesses the security of code generated by GitHub Copilot, a popular AI-based code assistant. We evaluate the security of code generated by Copilot across a range of security-related tasks. Our results show that Copilot-generated code is often insecure, particularly in the areas of authentication, authorization, and data handling. The study also identified several common security weaknesses in the generated code, such as hard-coded credentials, weak password requirements, and lack of input validation. The study also identified several common security weaknesses in the generated code, such as hard-coded credentials, weak password requirements, and lack of input validation.

CCS CONCEPTS
Security and encryption → Software security technologies → Security and privacy → Software security engineering

KEYWORDS
Code Generation, Security Weaknesses, LLMs, GitHub Copilot

ACM Reference Format:
Hammadreyya Purohit, Balogh Ahmad, Benjamin Tan, Brandon Dolan-Gold, Ramesh Korthi. 2021. Asleep at the Keyboard? Assessing the Security of GitHub Copilot’s Code Contributions. In Proceedings of the ACM Conference on Computer Security (CCS ’21), New York, NY, USA, 10 pages. https://doi.org/10.1145/3622383

1 INTRODUCTION
There is a burgeoning interest in designing AI-based assistants to assist humans in designing computer systems. However, such assistants generate code that is not always secure. This paper assesses the security of code generated by GitHub Copilot, a popular AI-based code assistant. We evaluate the security of code generated by Copilot across a range of security-related tasks. Our results show that Copilot-generated code is often insecure, particularly in the areas of authentication, authorization, and data handling. The study also identified several common security weaknesses in the generated code, such as hard-coded credentials, weak password requirements, and lack of input validation. The study also identified several common security weaknesses in the generated code, such as hard-coded credentials, weak password requirements, and lack of input validation.

arXiv:2308.02178v3 [cs.SE] 10 Aug 2023

Who Answers It Better? An In-Depth Analysis of ChatGPT and Stack Overflow Answers to Software Engineering Questions

Samia Bahir, David N. Udo-Iroch, Tianyi Zhang
Purdue University, Purdue University, Purdue University
West Lafayette, IN, USA, West Lafayette, IN, USA, West Lafayette, IN, USA
sbahir@purdue.edu, udorochd@purdue.edu, zhangtianyi@purdue.edu

ABSTRACT
Stack Overflow is a popular online community where developers ask and answer questions about software engineering. ChatGPT is a popular AI-based chatbot that can answer questions about software engineering. This paper compares the quality of answers provided by ChatGPT and Stack Overflow to software engineering questions. The study shows that ChatGPT answers are often more complete and more detailed than Stack Overflow answers. The study also shows that ChatGPT answers are often more accurate than Stack Overflow answers. The study also shows that ChatGPT answers are often more helpful than Stack Overflow answers.

CCS CONCEPTS
Security and encryption → Software security technologies → Security and privacy → Software security engineering

KEYWORDS
Code Generation, Security Weaknesses, LLMs, GitHub Copilot

ACM Reference Format:
Samia Bahir, David N. Udo-Iroch, Tianyi Zhang. 2023. Who Answers It Better? An In-Depth Analysis of ChatGPT and Stack Overflow Answers to Software Engineering Questions. In Proceedings of the ACM Conference on Computer Security (CCS ’23), New York, NY, USA, 10 pages. https://doi.org/10.1145/3622383

1 INTRODUCTION
Stack Overflow is a popular online community where developers ask and answer questions about software engineering. ChatGPT is a popular AI-based chatbot that can answer questions about software engineering. This paper compares the quality of answers provided by ChatGPT and Stack Overflow to software engineering questions. The study shows that ChatGPT answers are often more complete and more detailed than Stack Overflow answers. The study also shows that ChatGPT answers are often more accurate than Stack Overflow answers. The study also shows that ChatGPT answers are often more helpful than Stack Overflow answers.

introducimos inadvertidamente
estas vulnerabilidades en nuestra
vida cotidiana



Whitepaper

Veracode Fix: AI Code Remediation Done Right

“With Veracode Fix, developers receive precise guidance, reducing the time and effort needed to secure applications.”

- Sebastian Wilke, Cybersecurity Manager, Banco Galicia



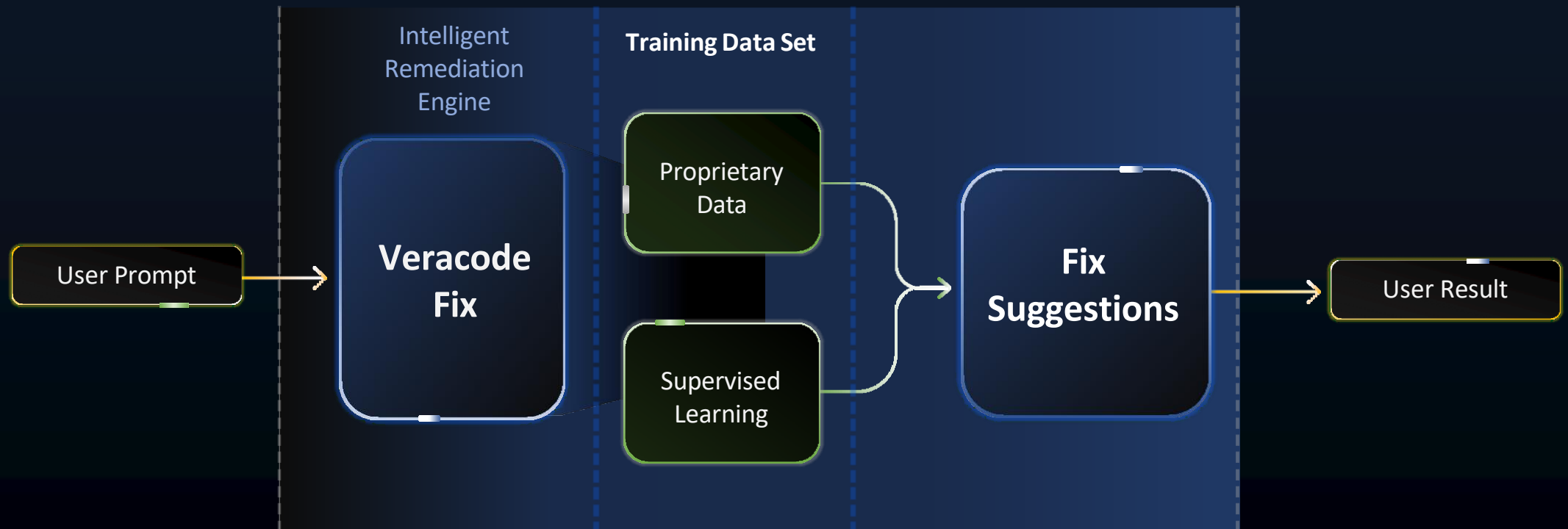
Veracode Fix Approach

Conjunto de datos de Veracode

Garantía de procedencia del código

Respaldado por la plataforma de seguridad de software inteligente de Veracode

Cobertura todo lo que importa

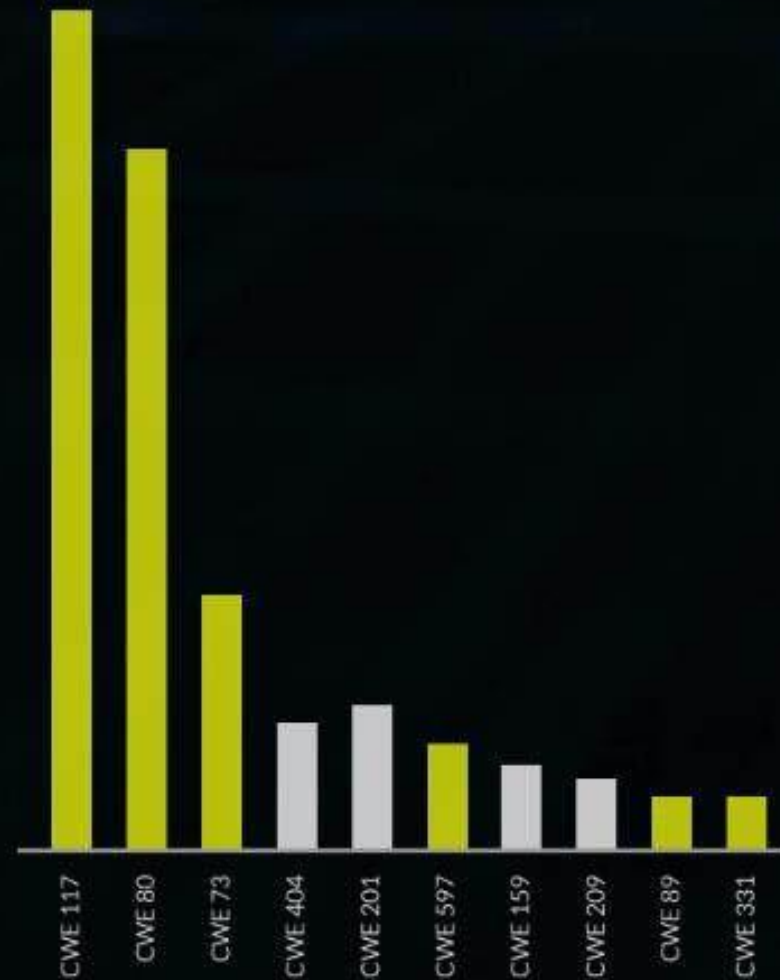


Efficacy of Veracode Fix Across Programming Languages



Percent of Veracode Static Analysis Java findings with Veracode Fix CWE coverage.

Top 10 Java CWE Static Analysis Findings

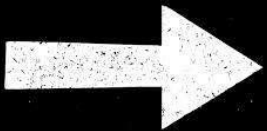




Veracode Fix revoluciona el proceso de remediación de vulnerabilidades, aprovechando el poder de la IA Generativa, para generar correcciones automáticas de fallos de seguridad.



Al entrenar el modelo GPT con datos cuidadosamente analizados y utilizando una amplia base de datos de parches de referencia escritos por nuestros investigadores de seguridad, Veracode Fix se convierte en una herramienta segura y eficiente.



Este enfoque mejorado con IA reduce reduce significativamente el tiempo de remediación de vulnerabilidades y por lo tanto la deuda técnica de seguridad

VERACODE

Application Risk Management Platform

Empowering the application security professionals and developers that create secure software for a safer world.

01

Use Cases



Secure
Your SDLC



Protect
Your Software
Supply Chain



Remediate
Risk

Stakeholders

Developer

Dev Ops

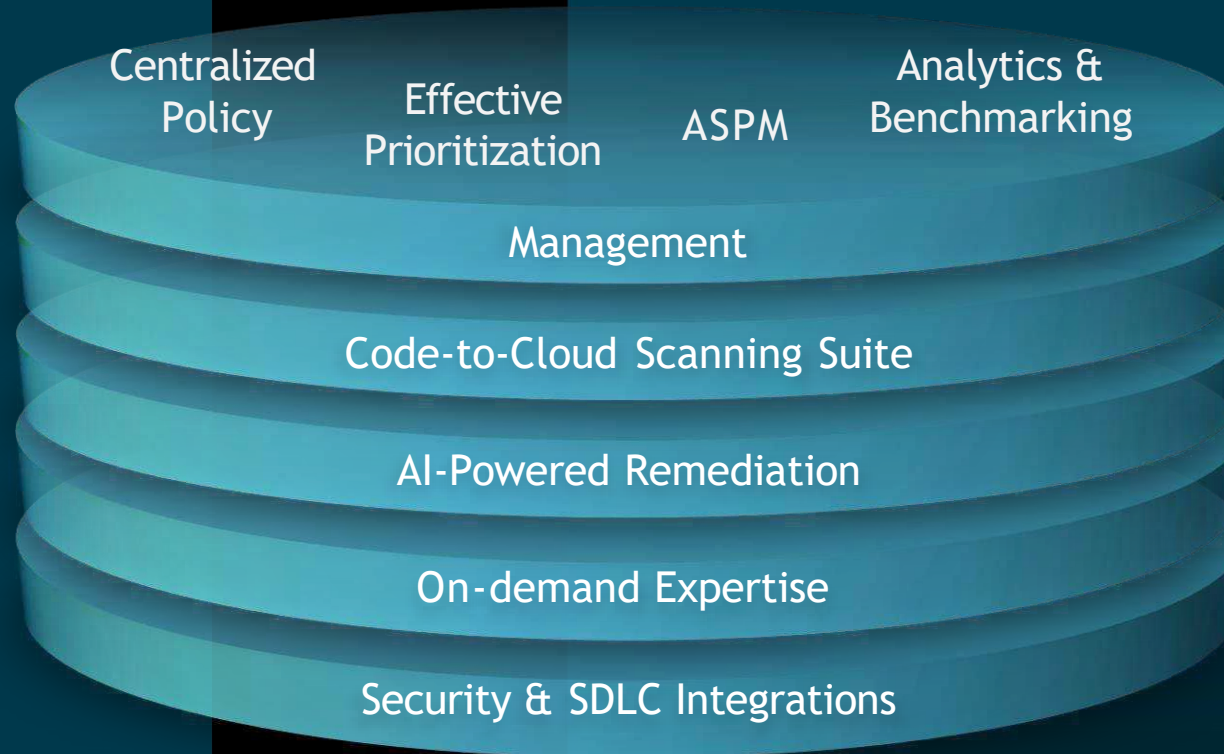
App Sec

Cloud Eng

CISO

Architect

Application Risk Management Platform



Application Risk Management Platform



Fast Start and Scale

- Automatically onboard developers
- Find and fix flaws in minutes

Actionable Visibility

- Code to Cloud Scanning Suite
- Highly accurate findings focus prioritization

Real-Time Flaw Remediation

- Rapid reduction of security debt
- Proactive flaw prevention

Securing The Code Pipeline

Continuously Find and Fix Flaws at Every Stage of The Modern Software Development Lifecycle

Veracode Intelligent Software Security Platform

Policy Management

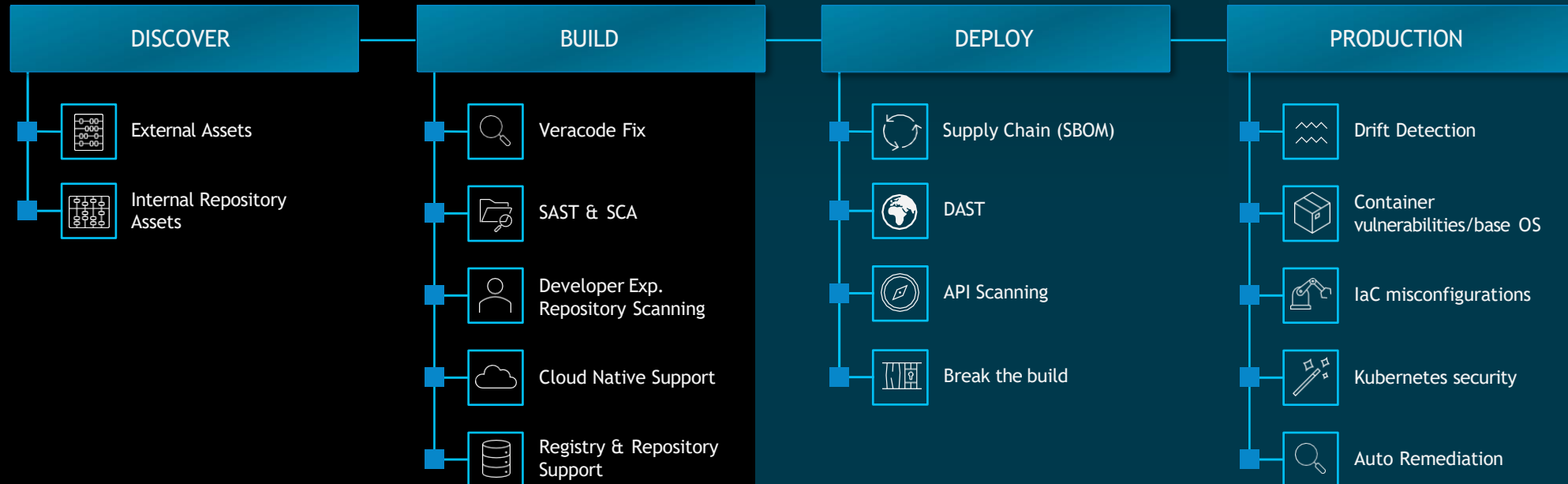
Reporting, Analytics & Peer Benchmarking

Data residency EU, FedRAMP

Developer & Security Tool integrations

Governance Risk & Compliance

Open API accessibility



A close-up photograph of a highly detailed, metallic robotic hand on the left, reaching out to grasp a human hand on the right. The robotic hand is silver and gold, with visible joints and wiring. The human hand is a natural skin tone. The background is solid black. The word "Gracias" is centered in white text between the two hands.

Gracias

VERACOTIDE

¡Ahora te toca!



¿Tienes una idea de negocio en ciberseguridad, tu proyecto necesita de ciberseguridad?

¡Te ayudamos a emprender!



¡Apúntate a los próximos talleres de emprendimiento!



¡O solicita plaza en el programa de Incubación de INCIBE Emprende!

Ediciones anteriores



INCIBE Emprende

¡Te estamos esperando!



proyectos@clubdeemprendimiento.com



www.clubdeemprendimiento.com





Programa de Impulso a la Industria de la Ciberseguridad Nacional

#INCIBEemprende