GIGATIC17

Más allá del futuro: Negocio, Tecnología y Robótica

26ABR2017_

itSMF ESPAÑA Catalunya

ISACA Trust in, and value from, information systems Barcelona Chapter

OWASP The Open Web Application Security Project

Técnicas de Evaluación de Seguridad en el Software

Vicente Aguilera Díaz

Sesión S01

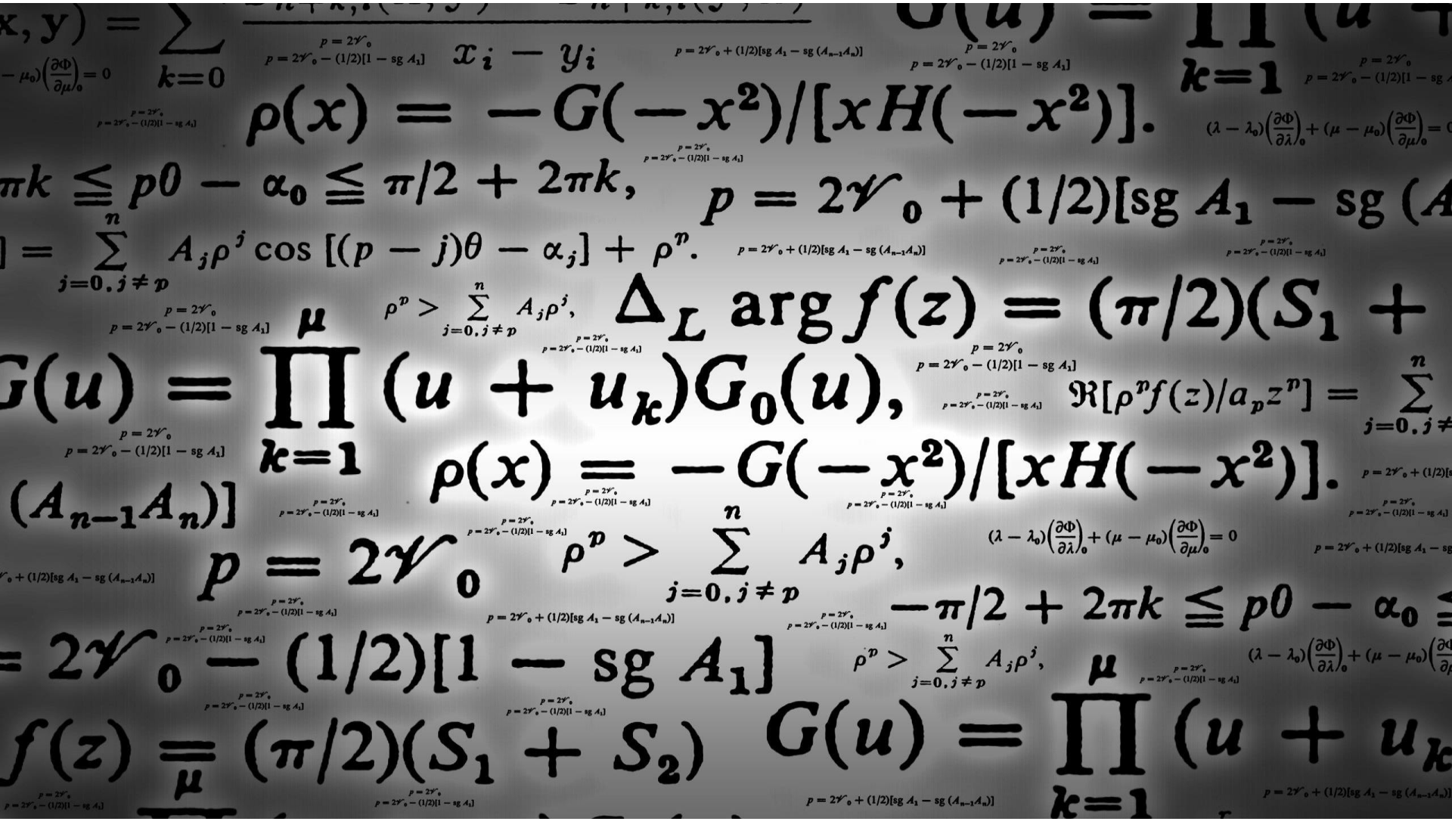
#gigaTIC17

Agenda

1. Complejidad del software
2. Crear software seguro
3. Técnicas clásicas de evaluación
4. Machine Learning aplicado al testing
5. Conclusiones

1. Complejidad del software

Técnicas de Evaluación de Seguridad en el Software



Técnicas de Evaluación de Seguridad en el Software

2. Crear software seguro

Software seguro

- Diseñar, construir y probar el software para la seguridad
- Continúa ejecutándose correctamente bajo un ataque malicioso
- Diseñado con el fallo en mente

Buenas prácticas

- OWASP SAMM
- OWASP CLASP
- Microsoft SDL
- SSE CMM
- Cigital Touchpoints
- BSIMM

3. Técnicas clásicas de evaluación

Técnicas de Evaluación de Seguridad en el Software

¿Qué es la evaluación?

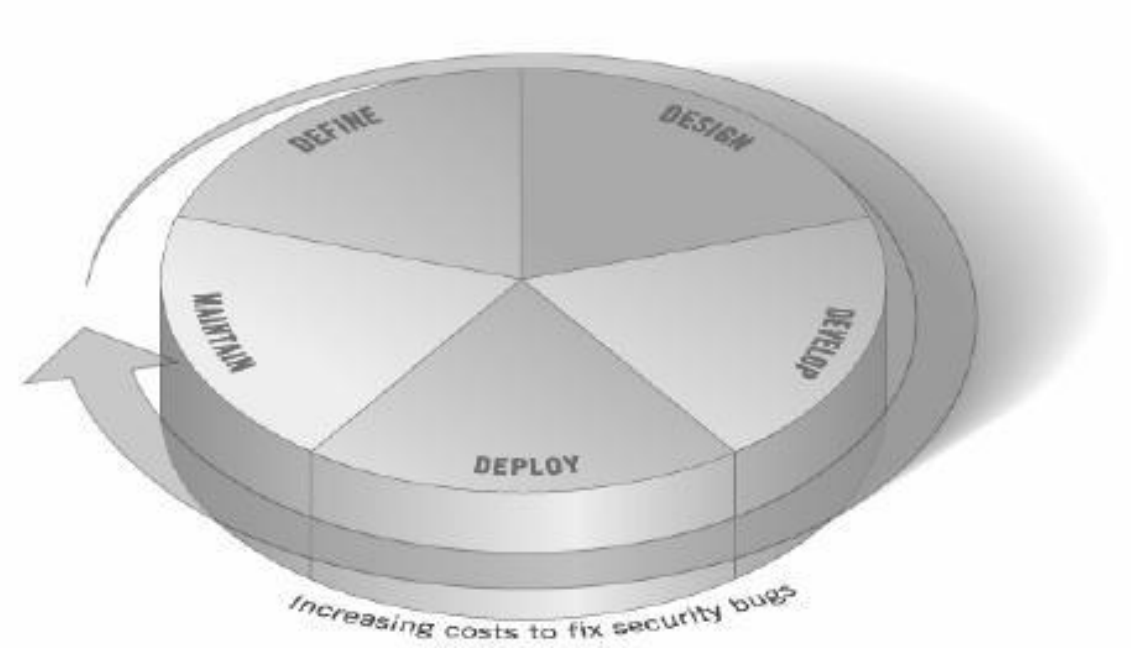
- Comparación del estado de algo con un conjunto de criterios

¿Porqué hacer la evaluación?

- Identificar gap entre las prácticas de la organización y las mejores prácticas de la industria

¿Cuándo hacer la evaluación?

- A través del SDLC



¿Qué incluir en la evaluación?

- Pesonas, procesos y tecnología

Técnicas clásicas

- Revisiones e inspecciones manuales
- Modelado de amenazas
- Revisiones de código
- Pruebas de intrusión

Revisiones e inspecciones manuales

- Entrevistas
- Análisis de documentación
- Evalúa el SDLC

Ventajas

- No requiere soporte de tecnología
- Flexible
- Promueve el trabajo en equipo
- Temprano en el SDLC

Inconvenientes

- Puede consumir mucho tiempo
- El material no siempre está disponible
- Require habilidades específicas para ser efectivo

Modelado de amenazas

- Descomponer la aplicación
- Definir y clasificar los activos
- Identificar vulnerabilidades potenciales
- Identificar amenazas potenciales
- Crear estrategias de mitigación

Ventajas

- Punto de vista práctico del atacante
- Permite tomar decisiones informadas sobre los riesgos de seguridad
- Flexible
- Temprano en el SDLC

Inconvenientes

- Require conocimiento profundo del software
- Es un proceso continuo y requiere actualización

Revisiones de código

- Búsqueda de vulnerabilidades en el código fuente de la aplicación
- Conocimiento real de lo que hace la aplicación

Ventajas

- Completo y efectivo
- Preciso

Inconvenientes

- Requiere altos conocimientos de desarrollo y seguridad
- No puede detectar errores en tiempo de ejecución
- El código fuente desplegado puede ser distinto del código analizado

Pruebas de intrusión

- Búsqueda de vulnerabilidades en tiempo de ejecución
- Simula el escenario de un atacante
- Explotación de las vulnerabilidades

"If you fail a penetration test you know you have a very bad problem indeed. If you pass a penetration test you do not know that you don't have a very bad problem"

Ventajas

- Puede ser rápido/económico
- Requiere “menos” habilidades que la revision de código
- Evalúa el código que está expuesto

Inconvenientes

- Demasiado tarde en el SDLC
- Sólo evalúa la funcionalidad accesible

4. Machine Learning aplicado al testing

Machine Learning

- *Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed.*

Aplicado a la evaluación del software

Description of the job

Airbus (Saint-Martin) is looking for an intern for a 6-month internship to work on leveraging system feedback within security testing through machine learning technics.

This internship will start in January/February 2017 (subject to some flexibility).

Aplicado a la evaluación del software

- DeepCode
- Creado por DARPA
- Analiza código fuente y binario
- Identifica la forma del código vulnerable

DEEPCODE

"Ultimately, the goal of DeepCode is to find all instances of all known software bugs"

Aplicado a la evaluación del software

- Code Phage
- Creado por MIT
- Identifica vulnerabilidades
- Es capaz de auto parchear el código
  - Entre 2-10 minutos

Aplicado a la evaluación del software

- Infer
- Facebook
- Análisis estático
- Open-source

Aplicado a la evaluación del software

- Diferencia con escáneres automáticos
- Escáneres automáticos
  - Identifican sólo vulnerabilidades conocidas
  - Elevada tasa de falsos positivos
  - Recomendaciones generales

Aplicado a la evaluación del software

- Detección de vulnerabilidades
  - A tiempo real
  - Optimización de las pruebas de seguridad
  - Prevención de nuevos vectores de ataque
  - Recomendaciones y aplicación de soluciones
- Elaboración de informes de evaluación
- Aportará reducción de costes
- Incrementará la calidad del análisis

5. Conclusiones

Conclusiones

- Construir software seguro implica adoptar un modelo de madurez
- Las pruebas deben servir para verificar y no para conocer el nivel de seguridad
- Aunque el uso de herramientas es necesario, siempre se requiere una revisión humana
- Es necesario entender el contexto del análisis y ser imaginativo en las pruebas

Conclusiones

- Las técnicas de evaluación clásicas siguen siendo válidas y necesarias
- Machine learning no será un sustituto, será un proceso complementario
  - Análisis estático inteligente
  - Automatización de pruebas de seguridad
- El mercado demanda conocimientos en inteligencia artificial y técnicas de machine learning

Referencias

Referencias

- Facebook AI tool squashing bugs is now open

  https://www.wired.com/2015/06/facebooks-ai-tool-squashing-bugs-now-open/
- Infer

  http://fbinfer.com
- DeepCode

  http://www.deepcode.ai

Referencias

- Code Phage - Automatic Error Elimination

  https://people.csail.mit.edu/stelios/papers/codephage_pldi15.pdf
- OWASP

  https://www.owasp.org
- OWASP SAMM

  https://www.owasp.org/index.php/OWASP_SAMM_Project

Técnicas de Evaluación de Seguridad en el Software