Introducción a la Seguridad

Prof. Ernesto Rodriguez

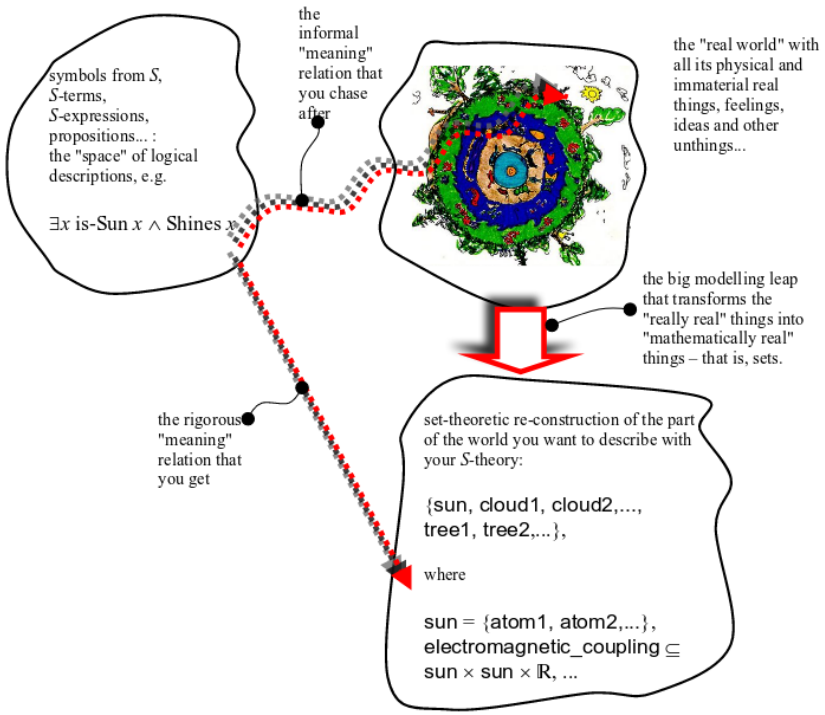
Universidad del Itsmo

erodriguez@unis.edu.gt

Motivación

- Un sistema de software procesa, almacena y recupera datos de diversas fuentes según un conjunto de reglas.
- Muchas veces dichas reglas permiten acciones no deseadas, ya sea en el sistema mismo o en la computadora que corre dicho sistema.
- Es dificil traducir reglas del mundo real a reglas que puede entender una computadora.
- Los sistemas complejos estan construidos sobre software escrito por otras personas, cada programa que se utiliza para crear un sistema puede tener funciones no esperadas.
- Recordemos el *Teorema de Rice*[1]

Seguridad: Vista general



- Dado un conjunto de reglas ϕ de un sistema
- Supongamos que ψ expresa una utilización indevida del sistema.
- Si $\phi \models \psi$, entonces nos pueden hakear.

Seguridad: ¿Que puede salir mal?

¡Muchas cosas!, entre ellas:

- Los datos se modifican de forma indevida.
- Un usuario obtiene acesso a datos que no puede ver.
- Un usuario escala sus privilegios en un sistema.
- Un usuario falsifica un documento.

En resumen, a pesar que la criptografia y otros mecanismos son robustos, es muy dificil utilizarlas correctamente.

Seguridad: Injección de SQL y Ataques de XSS

- Un sitio web, app u otros solicita información al usuario (ie. una busqueda).
- Esta información luego se almacena en la base de datos.
- El sistema genera una solicitud a la base de datos con los datos ingresados *sin hacer ningúna validación*
- Los datos contienen codigo, ya sea SQL o Javascript.
- Sucede alguno de:
  - El codigo SQL solicita datos de la base de datos.
  - El codigo SQL hace cambios en la base de datos.
  - El codigo Javascript luego se presenta y ejecuta en paginas de otros ususarios.

Seguridad: Errores

Errores:

- **Error #1:** Confiar en los datos del usuario
- **Error #2:** No almacenar los datos privados de forma segura
- **Error #3:** Separación inapropiada de datos y codigo

Observaciones:

- Este error fue muy comun en PHP. No por que PHP fuera inseguro como tal, sino que permitia que personas con poco conocimiento levantaran sitios web.
- La mayoria de los *Web Frameworks* y *ORMs* modernos automaticamente protegen la base de datos de estos errores asi que más vale utilizar alguno.

Seguridad: Buffer Overflows

- Un sistema obtiene datos (generalmente un string) de un lugar externo al sistema.
- El sistema no valida la longitud ni contenido del string de forma adecuada.
- El sistema copia el string a un *buffer* en la memoria, y no revisa el tamaño del buffer.
- Por lo general se utiliza una funcion como `strcpy`, la cual no valida si el buffer tiene la capacidad necesaria.
- Al exceder la capacidad del buffer, parte del contenido del string se copia a la memoria del programa. Este contenido por lo general tiene codigo ejecutable.
- El comportamiento del programa cambia debido a que su memoria ahora tiene codigo introducido del exterior.

Seguridad: Errores

- **Error #1:** Confiar en datos externos al sistema
- **Error #2:** Separación incorrecta de datos y codigo
- **Error #3:** El sistema se ejecuta con privilegios elevados.

Observaciones:

- Estas vulnerabilidades por lo general solo se dan en lenguajes de bajo nivel como C y C++.
- Muchos sistemas operativos y procesadores tienen mecanismos que permiten proteger la memoria contra modificaciones indebidas.
- ¿Que patrones se repiten?

Referencias


Wikipedia.
Rice's theorem.
https://en.wikipedia.org/wiki/Rice%27s_theorem.