Nos mostro un articulo que dice que tenemos una dependencia excesiva a librerias open-source para programar y confiamos absolutamente en su integridad. Es dificil armar un programa desde cero sin usar muchas librerias. El autor busca hacer una pieza de software "lean" usando la menor cantidad de librerias posibles.

Criptografía y Seguridad

Seguridad en aplicaciones: Flujo de información

Motivación

- Escenario: permitir a los docentes escribir los exámenes e impedir que los alumnos accedan a los mismos
- ACLs:
  - ACL(/var/cys/examenes): { (pablo, r), (ana, r) }
- ¿Sirve el control de acceso como mecanismo efectivo?

La respuesta es que sirve pero desde una version muy estatica, cuando no hay ningun cambio en el sistema.
Es decir dejamos de lado como lecturas, escrituras, ejecuciones, etc.

Motivación

- Escenario: permitir a los docentes escribir los exámenes e impedir que los alumnos accedan a los mismos
- ACLs:
  - ACL(/var/cys/examenes): { (pablo, r), (ana, r) }
  - ACL(/tmp): { (pablo, rw), (ana, rw), (juan, rw) }
- ¿Que ocurre si un editor de textos guarda una copia en /tmp mientras trabaja con un documento?

La idea es que todo el tiempo hay FLUJOS de informacion.
Por lo tanto, los ACL se deben implementar sobre los flujos de informacion y no solo sobre la informacion en si

Políticas y ACLs

- Las políticas, por lo general restringen el flujo de información y no el acceso a objetos
- Comparar:
  - Evitar que un empleado sepa el sueldo de otro

    Vs
  - Evitar que un empleado acceda a la base de datos de sueldos
- ¿Entonces los ACLs no sirven?
  - Sirven, pero por lo general son mecanismos abiertos.
  - Deben ser complementados

Control de acceso vs flujo

- Control de acceso
  - Limita acceso a operaciones sobre objetos
  - Los objetos contienen información
    - Limita el acceso a información
- Pero
  - La información no es estática
    - Es actualizada
    - Puede copiarse

Entropía

- Definida sobre una variable aleatoria discreta
- Sea X una V.A.D. Que toma valores x_1 … x_n

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i)$$

- Mide incertidumbre a la hora de determinar el valor de una variable
- Maximo: $p(x_i) = 1/n$
  - Distribución uniforme
- Minimo: $p(x_i) = 1$, $p(x_j) = 0$ para $j \neq i$
  - Evento único

Define informacion desde un punto de vista matematico.
Nosotros tenemos una variable aleatoria discreta X que puede tomar distintos valores.
Se define como la informacion de la variable aleatoria como menos el locaritmo de la probabilidad de cada posible valor que puede tener esa VAD.
La entropia es el promedio ponderado por probabilidad de cada posible valor. Es una medida de la aleatoriedad de todas las posibilidades de un sistema.

Cuando la entropia es cero, hay un unico evento. esto significa que NO hay informacion.

Entropía condicional

- Sea X una V.A.D. Que toma valores $x_1 \dots x_n$
- Sea Y una V.A.D. Que toma valores $y_1 \dots y_m$

Cuando la entropia es cero, entonces el flujo de informacion es siempre el mismo, entonces no hay "nada nuevo", entonces NO hay informacion.

$$H(X|Y=y) = -\sum_{i=1}^{n} p(x_i|y) \log p(x_i|y)$$

Si yo se que el evento "Y" ocurrio, me interesa averiguar las probabilidades de que el evento "X" ocurra. Esto se llama entropia condicional.
Esto nos ayuda a definir la idea de "flujo de informacion".

$$H(X|Y) = \sum_{j=1}^{m} p(y_j) H(X|Y=y_j)$$

Flujo de información: Definición

- Sea s el estado de un sistema
- Sea t el estado del sistema luego de ejecutar comandos $c_1, \ldots c_n$
- Sean x_s, y_s valores de objetos en el estado s
- Sea y_t valor de y en el estado t
- Hay flujo de información de x a y si:

y_t = valor de y en el estado t

- $H(x_s \mid y_t) < H(x_s \mid y_s)$, si y existe en el estado s
- $H(x_s \mid y_t) < H(x_s)$, si y no existe en el estado s

Estas ecuaciones buscan ver que el hecho de que ocurra un evento no nos aporte informacion extra en cuanto a la ocurrencia de otro evento.
Si esto pasa, hay flujo de informacion.

Hay flujo de informacion si la entropia que yo tenia en mi sistema original se redujo.
La idea es que si se reduce la entropia, entonces se puede inferir mas informacion que antes.

Seguimiento de flujo de información

- Existen técnicas formales de análisis de flujo de información en programas.
- En general existe flujo de información en un programa.
    - Es necesario para funcionar
- A veces interesa verificar casos puntuales
    - Que una función de cifrado no revele información de la clave

Ejemplo

- Considerar el comando:
  - y := x + z
- Donde
  - 0 <= x <= 7, con igual probabilidad (los valores de z no son equiprobables)
  - Z= { p(z=1)=0.5, p(z=2)=0.25, p(z=3)=0.25 }
- Entonces
  - H(x) = - 8 ∑ (1/8 lg (1/8)) = 3

Calculamos la entropia de X. Nos da 3 porque tenemos 8 valores posibles, que se pueden expresar con 3 bits. Notar que esto solo se da cuando la distribucion es uniforme.

Hay traspaso de información de x a y!

- Conociendo y luego del comando
  - x puede tener solo 3 valores (y-1, y-2, y-3)
  - H(x | y) = –(1/2) lg (1/2) –2(1/4) lg (1/4) = 1.5

Hay flujo de informacion, porque una vez que conocemos Y, se reduce la entropia de X

Flujo indirecto

- La información puede fluir de manera indirecta
- Ejemplo:
  - If x = 0 then y = 1 else y = 0
- x e y no aparecen en la misma asignación
- Considerar x como {0,1} con p(x=0) = 0.5
  - H(x) = 1
  - H(x | y) = 0
  - Hay traspaso de información!

Flujo indirecto

- La información puede fluir por comportamiento
- Ejemplo:
  - while x = 0 loop {}

Aca nos estamos basando en si el programa termina o no termina

- No existe ninguna asignación
- Considerar x como {0,1} con p(x=0) = 0.5
  - Definir y = 0 si el programa termina
  - H(x) = 1
  - H(x | y) = 0
  - Hay traspaso de información!

Flujo de información

- Explicito
  - Existe una asignación o escritura de información del tipo y:= f(x)
- Implícito
  - Verificación de flujo de información sin asignaciones explicitas
- **Problema**: encontrar e controlar los flujos implícitos de información

Requerimientos

- Toda política de control de flujo de información debe cumplir con dos principios:
- Reflexión
  - a, b en la misma clase
  - a puede leer/escribir o1
  - → b puede leer/escribir o1
- Transitividad
  - a, b en diferentes clases
  - a puede escribir o1 y leer o2
  - b puede leer o1
  - → b puede leer o2

Flujo de información - Bell LaPadula

- Si a, b ϵ C_1, y o ϵ C_1, tanto a, b pueden leer y escribir o porque:
  - A dom o y o dom a
  - B dom o y o dom b
- Si a ϵ C_1, b ϵ C_2 dom C_1, o ϵ C_1, b puede leer o
  - Como b dom a, a dom o → b dom o
- Más complejo con restricciones discrecionales
  - Si esta en curso la restricción (b, -, C_1)
  - A puede leer o y escribir o' en C_2, que b puede leer

Mecanismos

- Estáticos
  - Se analiza el flujo de información comando por comando
    Tiene que ver mucho con analisis estatico de codigo
  - Utiliza conceptos de teoría de compiladores
  - Solo se permiten comandos 'certificados'
- Dinámicos
  - Se asignan etiquetas a la información contenida
  - Al leer, un usuario adquiere la/s etiqueta/s
  - Al escribir, el dato lleva todas las etiquetas
  - Cada zona tiene un conjunto de etiquetas requeridas y prohibidas

Ejemplo

- En esferas gubernamentales, cuando un oficial adquiere derechos de acceso a información confidencial, deja de poder emitir comunicados oficiales públicos
- PERO, el control de información puede escapar del ámbito técnico
  - ¿Como impedir que hable informalmente?
  - ¿Como impedir que imprima documentación clasificada?
  - ¿Como impedir que fotografíe una pantalla?

El problema de confinamiento

- Sistema ideal:
  - Permite que una entidad acceda solo a los recursos para los cuales está autorizado
    - Parte “fácil” del problema
    - Existen mecanismos seguros
  - No revela información de ningún tipo a quien no está autorizado
    - Parte “dificil” del problema
- Problema de confinamiento:
  - Prevenir que un servidor revele información que el usuario del servicio considere confidencial

Aislación total

- Requisitos:
  - El proceso no puede comunicarse con otros procesos
  - El proceso no puede ser observado
- Consecuencias:
  - El proceso no revela información
  - En la práctica, los procesos usan recursos medibles:
    - Memoria
    - Ciclos de CPU
    - Espacio en disco
    - Ancho de banda

La idea es que es muy dificil que un proceso este TOTALMENTE aislado.

Ejemplo

- Los procesos a y b no pueden comunicarse
  - Pero ambos comparten el sistema de archivos
  - Si no comparten el sistema de archivos, comparten el procesador
  - Además comparten memoria
- Todos estos recursos son observables
  - Permiten crear un canal oculto de información

Canal oculto

- Un canal de comunicación que no fue diseñado para ello
- Clasificación:
  - Canal oculto espacial
    - Utiliza atributos de recursos compartidos
  - Canal oculto temporal
    - Utiliza información temporal o de orden en el acceso a recursos compartidos
- Atributos:
  - Ruido: capacidad de interferencia no premeditada de terceras partes
  - Ancho de banda: tasa de trasmisión

Ejemplo

- Dos procesos no pueden comunicarse, pero corren en el mismo servidor
- Comparten la CPU, entonces hay un posible canal oculto:
    - Para enviar un bit 0, el proceso a devuelve el control al SO inmediatamente
    - Para enviar un bit 1, el proceso hace uso intensivo de su slot temporal
    - El proceso b accede al reloj de tiempo real y mide el tiempo hasta volver a tener control de la CPU

Ahora se esta hablando mucho de generacion de imagenes por senales wifi. Esto seria un canal oculto.

Side channel attacks

- Hacen uso de un canal oculto para ganar información
- Considerar el algoritmo para calcular $a^b \bmod n$:

```
x := 1; atmp := a;
for i := 0 to k-1 do begin
  if b_i = 1 then
     x := (x * atmp) mod n;
  atmp := (atmp * atmp) mod n;
end
result := x;
```

b_i = iésimo bit de la representación binaria de b

El tiempo de ejecución depende de los bits de b.
Aplicando métodos estadísticos, se puede reconstruir parte de b

Métodos de aislación

- Presentar un ambiente que se comporte como una computadora que solo corre los procesos aislados
  - Es el concepto de máquinas virtuales
  - No requiere modificar los sistemas
- Correr los procesos en un ambiente que analiza las acciones y detecta fugas de información
  - Es el concepto de Sandbox
  - Requiere modificar los sistemas

Maquinas virtuales

- Son programas que simulan el hardware de una maquina
  - Puede ser una maquina real o una abstracta
  - Permite correr a los sistemas operativos sin modificación
  - El núcleo de la maquina virtual se convierte en el agente que provee seguridad
    - Los sujetos son las máquinas virtuales
    - Los objetos son son los recursos
- Ejemplos:
  - KVM, VmWare, qemu, CCS64, Mame, etc
  - Java virtual machine

Sandboxes

- Forman un ambiente donde las acciones están limitadas de acuerdo a cierta política
- Dos formas de trabajo:
  - Se modifica el ambiente
    - Los programas no deben modificarse
    - El kernel / SO es modificado para imponer las restricciones
  - Se modifica el programa
    - Se agregan llamadas a puntos de control
- Ejemplos:
  - Chroot, Gentoo ebuild sandbox
  - Java virtual machine (!) via SecurityManager

Lectura recomendada

Capítulo 16-1
Capítulo 17
Computer Security Art and Science

Matt Bishop