

Pregunta 1

10 de 10 puntos



La manera más segura de evitar que un sistema de APIs interna de un Sistema Operativo sea vulnerado es mantener secreta la documentación.

Respuesta seleccionada: ☒ Falso
Respuestas: ☐ Verdadero
☒ Falso

Comentarios para respuesta: Eso sería seguridad por ofuscación.

Pregunta 2

10 de 10 puntos



Si pudiéramos calcular la entropía de los sonidos, comparando entre ruidos de una ciudad y ruidos de un bosque. ¿Cuál cree que tendría una entropía más alta y por qué?

Respuesta seleccionada: El del bosque.
En la ciudad hay ruido constante por lo que los bytes de los sonidos de la ciudad estarían casi todos con bits de manera uniforme. (casi todos en 1)
En cambio, si grabamos los sonidos en un bosque, no siempre hay ruido sino que ocurren cada intervalos (algun animal, alguna tormenta, etc) por lo que los bytes, en general, serían mucho más monótonos.... excepto en estos "picos" en los que ocurre algún evento.
Esto hace que, los bytes de la ciudad sean más uniformes que los bytes del bosque, haciendo que la entropía más alta ocurra en el dataset del bosque.

Respuesta correcta: [None]

Comentarios para respuesta: Yo creo que la entropía sería mayor la de la ciudad. Hay más ruido constantemente ! Más ruido, más azar.

Pero la manera en la que lo argumentas es correcta desde el lado de entender la idea de entropía.

Pregunta 3

7 de 10 puntos



Un ex-compañero graduado le comenta sobre su startup de un producto embebido. Luego de detallar los aspectos técnicos, le explica como anécdota que tuvo que aprender a deshabilitar controles de canary y de ejecución de stack y heap para el binario final por unas funcionalidades extras que requería. ¿Podría esto ser un problema? En caso de que sí, explicar por qué. En caso de que no, justificar por qué no.

Respuesta seleccionada: Sí, podría ser un problema ya que viola uno de los principios de diseño: **Valores iniciales**.
Al modificar aspectos "estándar" del código/producto, como pueden ser los controles de canary y de ejecución de stack y heap, no solo puede traer conflictos con otros programas externos con los que interactúa sino que también puede causar vulnerabilidades por no contar con dichos mecanismos.
(Detalle agregado: al no seguir los lineamientos de los valores iniciales, ocurre un flujo de información también.... si todos los sistemas se acomodan a un estándar y viene uno que NO.... es por algo.)

Respuesta correcta: [None]

Comentarios para respuesta: Está bien lo que planteas pero es importante destacar que esos controles están implementados para evitar ataques de buffer overflow. Y que hoy eliminarlos implica de por sí arriesgarse muchísimo más, porque la media es algo mucho más alto ya, donde todos los sistemas lo implementan.

Pregunta 4

0 de 10 puntos



Considerar una política de contraseñas de 8 caracteres, entre ellas letras del alfabeto inglés en minúscula y números. ¿Cuál sería la probabilidad de que un atacante pueda obtener la contraseña en menos de un año si este contase con una tasa de pruebas de 23.000 pruebas por segundo y si se cuenta con 5 bits de salt?

Respuesta seleccionada: espacio de claves: $[a-z] + [0-9] = 26 + 10 = 36$
tiempo de la prueba = 365 días = $365 \cdot 24 \cdot 60 \cdot 60 = 31536000$ segundos
tasa de pruebas = 23000 x segundo
OBS: 5 bits de salt!!

Forma de la clave: _ _ _ _ _ (5 bits del salt) + _ _ _ _ _ _ _ _ (8 caracteres del espacio de claves)
==> Tamaño del espacio de claves = $36^5 \cdot 2^5 = 1934917632$

por lo tanto, utilizando la **Formula de Anderson**, puedo calcular la probabilidad de la siguiente manera:

$P = (\text{tiempo de la prueba} \cdot \text{tasa de pruebas}) / \text{Tamaño del espacio de claves} = (365 \cdot 24 \cdot 60 \cdot 60 \cdot 23000) / 36^5 \cdot 2^5$
==> $P = 374,86$ (bastante alta :/)

Respuesta correcta: [None]

Comentarios para respuesta: No ! Una P NUNCA PUEDE SER MAYOR QUE 1 !!!

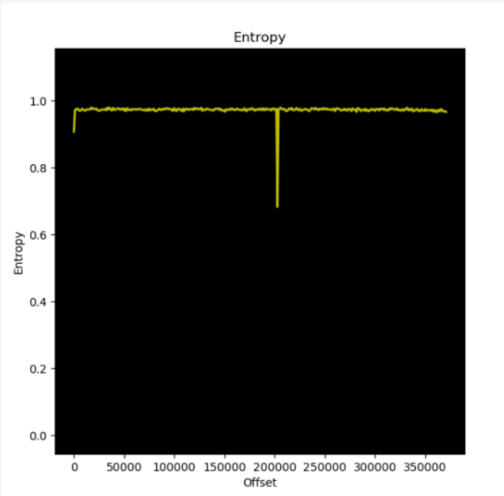
Además en rigor es un número muy pequeño ! Siempre que hagas una cuenta fíjate que tenga sentido lo que te da !

Pregunta 5

10 de 10 puntos



La siguiente imagen es el resultado de un análisis sobre una imagen adjunta en un mail sospechoso. En el eje X se presenta el offset en bytes desde el principio del archivo y el eje Y la entropía sobre ese byte. ¿Qué puede decir sobre el archivo? Fundamente por qué el sistema pudo detectarlo como sospechoso. ¿Qué representa la Entropía ?



Respuesta seleccionada: Lo que puedo asegurar al ver la imagen es que entre los bytes 200000 y 220000 (aproximadamente) los valores de los mismos, en comparacion a los valores que venian teniendo el resto de los bytes, cambian!! Se puede calcular la distribucion que siguen los bytes utilizando la entropia de la imagen... de esta forma, obtenemos una manera de analizar la imagen en su totalidad y ver si los bytes de la misma son "uniformes" o hay algunos con "anomalias". (como se los ve en este caso). Esto puede significar varias cosas, siendo una de ellas que en la imagen se encuentra esteganografiado un archivo/informacion entre los bytes 200000 y 220000 (aproximadamente).

(Detalle extra: yo pienso la entropia como el "balance de las cosas", bastante guiado por la definicion de entropia de fisica y la "entropia del universo")

Respuesta correcta: [None]

Comentarios para respuesta: Genial

Pregunta 6

10 de 10 puntos



Dado el siguiente snippet de código, indicar qué errores encuentra. En caso de encontrar algún bug, indicar por qué es un bug y qué input lo explotaría. (Ayuda: El número ASCII de las letras mayúsculas comienza en 0x41 = 'A').

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(){
5     int sectest = 0;
6     char buffer[10];
7
8     printf("User Login\n");
9     printf("Please insert password:\n");
10    fgets(buffer, 15, stdin); //extra space just in case
11    if(sectest==0x41424241){
12        printf("Access granted!\n");
13    }else{
14        printf("Keep trying %x\n", sectest);
15    }
16 }
```

Respuesta seleccionada: Puede ocurrir un buffer overflow! El buffer se lo definio de un tamano de 10 mientras que el fgets puede leer hasta 15.... Si como entrada le mandamos una cadena de caracteres que supere los 10 bytes, ocurrira un segmentation fault haciendo que el programa aborte como medida de seguridad para no comenzar a pisar memoria. De todas formas, si el programa fue compilado sin fsanitize / las verificaciones de que no suceda un pisado de memoria; se podria encontrar algun exploit probando distintos largos de cadenas de caracteres y analizando su comportamiento para deducir que memoria es la que se esta pisando, pudiendo asi modificarla por algo que el atacante desee. e.j. tal vez de forma contigua al buffer, se guarda la cadena de caracteres que hace TRUE al if (0x41424241 = 'ABBA'). De ser asi, podriamos enviar como cadena: 'TOMAS0 TOMAS', lo que guardaria en el buffer 'TOMAS' y pisaria la validacion contra 'ABBA' por una validacion contra 'TOMAS' (o cualquier otro ejemplo que se nos ocurra), lo que haria TRUE esa validacion, obteniendo como respuestas el "Access granted!" buscado.

Respuesta correcta: [None]

Comentarios para respuesta: [No se ha dado ninguna]

Pregunta 7

10 de 10 puntos



En un sistema de control industrial muy importante (equipos embebidos que controlan desde cintas transportadoras hasta bombas de gases tóxicos), existe una metodología para realizar los cambios. El encargado del equipo tiene una llave puesta en este en modo "RUN" y cuando quiere realizar un cambio, debe salir de su estación de trabajo, cambiar de edificio, ponerse las protecciones necesarias y girar la llave a modo "CHANGE", luego volver a la estación de trabajo, realizar el cambio y repetir el operativo para poner el sistema en modo "RUN" nuevamente. Para ahorrar tiempo, muchos operarios dejan la llave puesta en modo "CHANGE" e ignoran los carteles de alerta para que el sistema funcione igual. ¿Qué principios de diseños se están violando en esta situación? Tanto por parte del operario como por parte del diseñador del sistema. ¿Por qué?

Respuesta seleccionada: **Aceptacion psicologica y Economia de Mecanismos:** ya que si el sistema estuviese bien disenado para que pueda ser usado como corresponde, los usuarios del mismo cumplirian con las reglas y no sucederia lo que sucede en este caso. Como no lo esta, esto hace que los usuarios rompan con las reglas dejando la llave girada en modo "CHANGE", posiblemente generando una vulnerabilidad importante.

Respuesta correcta: [None]

Comentarios para respuesta: [No se ha dado ninguna]


Pregunta 8

10 de 10 puntos



Luego de que los usuarios se crean una cuenta en la aplicación "para mobile" de PagosMercado (pensada para personas de 18-80 años) el sistema no utiliza una contraseña para loguearse, sino que en cambio, utiliza autenticación biométrica utilizando la huella digital o un "pin" de 4-6 dígitos. ¿Qué principio de diseño diría que utilizaron los diseñadores a favor de este cambio? ¿Por qué? ¿Cuántos factores de autenticación está utilizando la aplicación?

Respuesta seleccionada: El principio de **Aceptacion Psicologica** ya que, haciendo que el sistema pueda loguear al usuario a traves de algo biometrico, reduce el "estres" que tiene el usuario de tener que acordarse de una contraseña al momento de iniciar sesion. Por lo mencionado en el enunciado, la aplicacion utiliza un UNICO factor de autenticacion: o el biometrico, o un "pin" de 4-6 digitos. (NO los 2) (**deberia usar los 2 ==> 'algo que se' + 'algo que tengo'**).

Respuesta correcta:  Se utiliza principalmente el principio de "Aceptación Psicológica" ya que para una persona de mayor edad es más fácil recordar un pin de 4 números en vez de una contraseña que cumpla con los estándares "comunes" (12 caracteres mínimos, letras, números, caracteres especiales, etc). La aplicación está requiriendo 2 mecanismos de autenticación, uno es el "algo que tengo" que sería mi celular y el otro es el "algo que sé" que es el pin.

Comentarios para respuesta: [No se ha dado ninguna]

Pregunta 9

1 de 10 puntos



Enumerar y explicar algunas de las técnicas utilizadas en los sistemas operativos modernos para intentar mitigar ataques de buffer overflow.

Respuesta seleccionada: - **Análisis estatico:** a traves de herramientas como cpp check, PVS o las ultimas versiones de los compiladores en si (entre otras), se analiza el codigo ANTES de que el mismo se pueda ejecutar, en busca de leaks de memoria (entre otras cosas).
- (posiblemente) memory managers mas "inteligentes"

Respuesta correcta: [None]

Comentarios para respuesta: Más que nada son técnicas que intentan reestructurar y controlar al propio proceso en memoria y. evitar que pueda ser alterado maliciosamente. Canary Stack, Address Layout Randomization, División entre código y dato de la memoria.

Todo lo que nombrás son técnicas para eso, pero que no dependen del sistema operativo.

Pregunta 10

2 de 10 puntos



Suponiendo el software del controlador de las barras de grafito de la central nuclear de Chernobyl, ¿Se puede asegurar que un sistema cómo este es seguro mediante algún mecanismo?

Respuesta seleccionada: No usaria la palabra "asegurar" ya que es muy difícil (o imposible) asegurar que un sistema sea seguro..... Mas alla de eso, se le pueden implementar multiples cosas para ayudar en la seguridad del mismo. Algunos ejemplos son:
- Listas de controles de acceso (ACLs) o Lista de Capacidades: definir en ellas quienes tienen que permiso (leer, escribir, ejecutar) sobre cosas del sistema. e.j. archivo de configuracion del controlador, quien lo puede leer y quien lo puede modificar.
- Alguna variante del modelo Bel-Lapadula: por ejemplo la del low-watermark, asignandole distintos niveles a los usuarios que representan el nivel de acceso dentro del sistema; de tener algun comportamiento "extrano", disminuirselo.

Respuesta correcta: [None]

Comentarios para respuesta: En algunos casos puntuales, de sistemas relativamente pequeños e hiper críticos podés utilizar métodos de verificación formal que te proveen una garantía de que el programa hace lo que dice hacer y eso está probado.

Esto es una pieza de software que procesa una entrada y genera una salida por lo que no es en sí mismo un tema de control de acceso, o política.