Pregunta 1

Una empresa está desarrollando un chip que puede insertarse bajo la piel para exponer información de cualquier tipo. Sus usos iniciales están asociados a información de contacto ante emergencias e información médica, pero el chip puede ser utilizado para almacenar cualquier tipo de información.

Para ello, el chip cuenta con 32 compartimentos de 1024 bytes. Cada compartimiento posee una clave de escritura y una clave de lectura. Las claves de escritura y lectura son diferentes para cada compartimiento y aleatorias, pero todos los chips tienen las mismas claves.

Cuando alguien “compra” uno de los 32 espacios del chip, obtiene el par de claves asociados para poder modificar ese compartimiento.

Para escribir el compartimiento, el chip debe recibir utilizando una transmisión por proximidad (NFC) una trama con el siguiente formato: Header (4 Bytes) || Size (2 Bytes) || Data (Variable)

Donde:

Header son 4 bytes con valor fijo compuesto por los valores ASCII de las letras “MTRX”

Size es un número entre 0 y 1024 indicando la cantidad de información a guardar

Data es la información a guardar.

Toda la trama viaja cifrada utilizando AES-GCM con la clave de escritura correspondiente.

Si se utilizó alguna de las claves de escritura válidas, los datos quedarán almacenados. Sino, se ignorará la operación.

Para leer un compartimiento, el chip debe recibir utilizando una transmisión por proximidad (NFC) una trama con el siguiente formato: Header (4 Bytes) || Offset (2 Bytes) || Size (2 Bytes)

Donde:

Header son 4 bytes con valor fijo compuesto por los valores ASCII de las letras “RCAL”

Offset es un número entre 0 y 1023 indicando desde qué posición leer en el compartimiento

Size es un número entre 1 y 1024 indicando cuánto leer

Toda la trama viaja cifrada utilizando AES-GCM con la clave de lectura correspondiente.

Si se utilizó alguna de las claves de lectura válidas, los datos serán retornados. Sino, se ignorará la operación.

NOTA: Ambas operaciones de lectura y escritura son tramas que viajan por el mismo canal. Parte del procesamiento del programa del chip es determinar de qué tipo de operación se trata la trama que está procesando.

a) Indicar paso a paso que debe hacer el programa dentro del chip para validar un pedido de escritura y ejecutarlo (1 pto.)

b) Indicar paso a paso que debe hacer el programa dentro del chip para validar un pedido de lectura y ejecutarlo (1 pto.)

c) Si no fuese posible utilizar el criptosistema AES-GCM, ¿qué características debe tener el criptosistema que lo reemplace para poder seguir aplicando el mismo procedimiento? (1 pto.)

Pregunta 2

Considerar el problema conocido como Private Information Retrieval (PIR), donde se busca que una parte consulte información de un servidor (imaginar un registro en una base de datos), sin que este último pueda saber cual es la información consultada.

Está demostrado que la única implementación con garantías absolutas de garantizar esa propiedad consiste en que el servidor envía en cada requerimiento la base de datos completa y que el cliente tome el registro que le interesa. Claramente esto no es muy práctico.

Una implementación de PIR con algunas restricciones consiste en contar con N servidores independientes, donde cada entrada de la base de datos se cifra utilizando un criptosistema de umbral con parámetros (k, N). Para guardar una entrada nueva, cada servidor almacena una sombra del secreto compartido.

Para recuperar una entrada, el cliente elige k servidores al azar y les solicita la sombra asociada a la entrada que quiere recuperar, reconstruyendo localmente el secreto.

a) ¿Por qué desde el punto de vista de un único servidor no es posible saber qué información fue solicitada? (1 pto.)

b) ¿Cuáles serían los valores mínimos de k y N para garantizar que el compromiso de dos servidores no afecte la privacidad de los requerimientos y la caída hasta 4

servidores no afecte la disponibilidad de la información en el modelo teórico? ¿Por qué? (1 pto.)

x) En la práctica, un servidor comprometido podría a partir de un requerimiento realizar él mismo los requerimientos por la cantidad de sombras necesarias para reconstruir el secreto y vulnerar el principio de privacidad. ¿Qué mecanismos implementaría para evitar que esto suceda (ser detallado, a nivel de que información exactamente enviaría/utilizaría y como la validaría)? (1 pto.)

Pregunta 3

Alegria S.A., una fábrica de juguetes, decide construir un canal de venta en línea. Para ello contrata los servicios de Comercialus S.A., otra empresa que brinda como servicio una plataforma de ecommerce completa.

Para poder utilizar la plataforma, Alegría debe integrar a su catálogo (para mantener actualizado el catálogo de productos publicados), su sistema de control de inventario (para mantener actualizada la disponibilidad de productos e ir actualizándose con cada venta generada en el sitio).

Para esas integraciones, Alegría decide crear un pequeño programa, que utilice el API de Comercialus y mantenga actualizado el catálogo. Para el inventario, este servicio expone a internet una URL que es llamada desde Comercialus cada vez que hay una venta o una devolución. El servicio ha sido desarrollado como una aplicación auto contenida que levanta su propio web server y corre en una de las computadoras de Alegría (Alegría es una empresa pequeña, no tiene ni equipo de IT, ni servidores, ni datacenter propio).

Comercialus, interesado en que a Alegria le vaya bien, pero preocupado por la situación de la integración, dado que no hay personal de IT dedicado a mantenerla y monitorearla, decide realizar un pentest sobre la solución para detectar problemas.

Siguiendo la metodología de hipótesis de falla, establecer 4 hipótesis de vulnerabilidades que tengan alta probabilidad de existir en el escenario descrito. Por cada una describir solamente la hipótesis y la prueba que realizaría para confirmar o refutar. (1 pto. cada una)

Ejercicio 1

a. Primero, el programa hace un $Vrfy_K(c,t)$ con la clave. Si el Vrfy da 1, se desencripta el mensaje. Luego, chequea que los 4 bytes del header corresponden a los valores ASCII de las letras "MTRx". Si todo esta bien, almacena los datos enviados siempre y cuando no excedan los 1024 bytes.

Observacion: la clave indica a que compartimento se intenta escribir

b. Primero, el programa hace un $Vrfy_K(c,t)$ con la clave. Si el Vrfy da 1, se desencripta el mensaje. Luego, chequea que los 4 bytes del header corresponden a los valores ASCII de las letras "RCAL". Finalmente, se devuelve lo que quiere leer el dueño, pasado el offset, siempre y cuando sea menor que 1024 bytes.

c. El cliente deberia mandar una firma del mensaje junto con una clave publica y luego encriptar todo bajo la clave privada premia. De esta forma, cada compartimento tendria asociada una clave publica, ademas de la privada. Entonces, usamos criptografia simetrica para confidencialidad y asimetrica para autenticidad.

Ejercicio 2

a. Un servidor no podría reconstruir el secreto (a menos que $k = 1$ pero no cumpliría PIA). Esto se debe a que se necesitan k servidores para construir el secreto.

b. Por un lado, k debe ser mayor estricto al numero de servidores que pueden estar comprometidos para que no puedan reconstruir el secreto $\Rightarrow k > 2$. Por otro lado, en caso de que se caigan 4 nodos, k debe ser menor estricto a $N - 4$ i.e N menos la cantidad de nodos caídos.

Rta. $2 < k < N - 4 \Rightarrow$ como mínimo $k = 3$ y $N = 7$

c.

Ejercicio 3

(1) Improper Authorization

Amenazas: tampering e information disclosure

Esto ocurre cuando el sistema no hace chequeos en cuanto a los accesos a los recursos. Un atacante podria ejecutar acciones como usuario anonimo para las cuales no tendria que tener permisos.

Pruebas:

Hacer un pedido directamente al servidor y ver si se actualiza la cantidad de productos. Hacer eso con todas las acciones que deben estar restringidas.

(2) Allocation of Resources without Limits or Throttling

Amenaza: denial of service

Esta vulnerabilidad se da cuando el sistema no controla la alocacion de recursos limitados como CPU, memoria, conexiones, etc. Un atacante podria enviar multiples pedidos hasta que el sistema se rompa.

Pruebas: en un ambiente de prueba

Ejecutar un programa que haga muchos requests a la API y ver si las toma y si se rompe el sistema.

(3) SQL Injection

Amenazas: tampering, spoofing e information disclosure

Esto ocurre cuando el sistema no hace los controles necesarios sobre los inputs de los usuarios. De ser asi, un atacante podria hacer modificaciones a la base de datos.

Pruebas:

En todos los compos de entrada, intentar obtener informacion de la base de datos mediante instrucciones SQL y ver si efectivamente nos devuelve la informacion.

(4) Buffer Overflow

Amenazas: tampering, information disclosure y denial of service

Esto se da cuando el sistema intenta copiar una cantidad de datos en un area que no es lo suficientemente grande para contenerlo. Entonces, un atacante podria ingresar en algun campo de entrada (Ej. nombre del producto) un texto lo suficientemente grande como para romper el sistema.

Pruebas: en un ambiente de prueba

En todos los compos de entrada, ingresar datos muy grandes y ver si crashea el sistema.