UNIVERSIDAD DEL VALLE DE GUATEMALA
Redes - CC3067
Sección 11
Ing. Miguel Novella Linares

Laboratorio 2 - Parte 2

Detección y corrección de errores

José Pablo Orellana 21970
Diego Alberto Leiva 21752

GUATEMALA, 01 de agosto del 2024

Descripción de la práctica y metodología utilizada

En este laboratorio se desarrollaron programas con el objetivo de poder transmitir y recibir mensajes utilizando dos algoritmos de detección y corrección de errores. Hamming y CRC-32 correspondientemente. Los programas se estructuraron en una arquitectura de capas que incluyen las siguientes.

- **Aplicación:**
  - Solicitar mensaje: El emisor solicita al usuario el mensaje que desea enviar. También solicita el algoritmo de detección y corrección de errores que se utilizará (Hamming o CRC-32).
  - Mostrar mensaje: El receptor muestra el mensaje decodificado. Si se detectaron errores que no pudieron ser corregidos, se debe indicar con un mensaje de error.

- **Presentación:**
  - Codificar mensaje: La capa de presentación convierte el mensaje a su representación binaria ASCII. Por ejemplo, el carácter 'A' se convierte a '01000001'.
  - Decodificar mensaje: Convierte los bits binarios de vuelta a caracteres ASCII si no se detectan errores. Si se detecta un error, se notifica a la capa de aplicación.

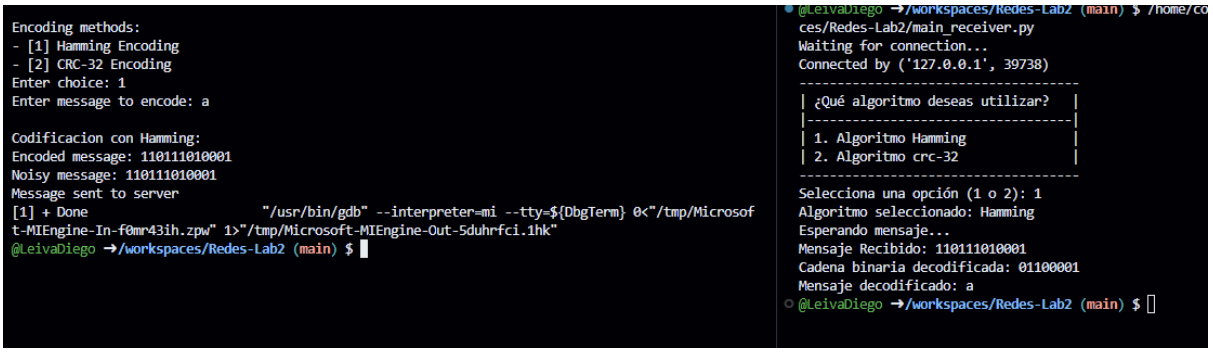
- **Enlace:**
  - Calcular integridad: Utilizando el algoritmo seleccionado (Hamming o CRC-32), se calcula la información de integridad y se concatena al mensaje binario original.
  - Verificar integridad: En el receptor, se recalcula la información de integridad y se compara con la proporcionada por el emisor para detectar posibles errores. Se informa a la capa de presentación sobre los resultados de la verificación.
  - Corregir mensaje: Si el algoritmo es capaz de corregir los errores detectados, esta capa realiza la corrección.

- **Ruido:**
  - Aplicar ruido: Para simular un canal no confiable, se introduce ruido en la trama binaria generada por la capa de enlace. La probabilidad de que cada bit se voltee está determinada por un parámetro de error (por ejemplo, 1/100).

- **Transmisión:**
  - Enviar información: Utilizando sockets TCP, el emisor envía la trama de información al receptor.
  - Recibir información: El receptor está en modo "server", escuchando en un puerto específico para recibir la trama de información.

Resultados

Hamming



CRC-32

Discusión

La implementación de los algoritmos de Hamming y CRC-32 nos permitió analizar sus capacidades y limitaciones en la detección y corrección de errores en la comunicación de datos. Es por ello que consideramos las siguientes ventajas y desventajas de cada uno.

Hamming

- **Ventajas:**
    - **Corrección de Errores de Un Solo Bit:** El algoritmo de Hamming (7,4) es capaz de corregir errores de un solo bit en cada bloque de 7 bits, lo que lo hace útil en entornos donde los errores aislados son comunes.

    - **Simplicidad y Eficiencia:** El algoritmo es relativamente simple de implementar y eficiente en términos de tiempo de cómputo.

- **Desventajas:**
    - **Limitación a Errores de Un Solo Bit:** Una de las principales limitaciones del algoritmo de Hamming es que solo puede corregir errores de un solo bit. Si se producen múltiples errores en un bloque de datos, el algoritmo puede no detectarlos correctamente o corregirlos de manera incorrecta.

CRC-32

- **Ventajas:**

    - **Alta Capacidad de Detección de Errores:** El CRC-32 es altamente efectivo en la detección de errores, especialmente para errores múltiples. La probabilidad de que un error pase desapercibido es extremadamente baja.

- **Desventajas:**

    - **No Corrige Errores:** A diferencia de Hamming, CRC-32 solo detecta errores pero no tiene la capacidad de corregirlos. Esto significa que, en caso de detectar un error, se debe retransmitir el mensaje completo.

Rendimiento y Eficiencia:

En términos de corrección de errores, Hamming ofrece una solución efectiva para errores de un solo bit con una baja sobrecarga de datos adicionales.
CRC-32, aunque no corrige errores, es mucho más robusto en la detección de múltiples errores, lo que lo hace ideal para transmisiones donde la integridad del mensaje es crítica.
Aplicabilidad:

Hamming es adecuado para entornos con baja probabilidad de error donde la corrección de errores es esencial y la simplicidad es una ventaja.
CRC-32 es preferible en escenarios con alta probabilidad de error y donde se prioriza la detección rápida de errores sobre la corrección.

Conclusiones

La elección del algoritmo depende de los requisitos específicos del sistema de comunicación. Hamming es ideal para sistemas con requisitos estrictos de corrección de errores de un solo bit y baja probabilidad de error, mientras que CRC-32 es más adecuado para sistemas donde la detección robusta de errores múltiples es crucial y la retransmisión es una opción viable.

Estas observaciones y comparaciones fueron validadas mediante pruebas exhaustivas, donde se simularon condiciones de transmisión con ruido y se analizaron los resultados obtenidos para ambos algoritmos.

Repositorio: https://github.com/LeivaDiego/Redes-Lab2

Referencias

Forouzan, B. A. (2007). Data Communications and Networking. McGraw-Hill.

Stallings, W. (2007). Data and Computer Communications. Pearson Prentice Hall.

"Hamming Code Error Detection and Correction", GeeksforGeeks. Disponible en: https://www.geeksforgeeks.org/hamming-code-error-detection-and-correction/

"CRC Calculation", Barr Group. Disponible en: https://barrgroup.com/Embedded-Systems/How-To/CRC-Calculation-C-Code