Científica, vol. 22, núm. 1, pp. 3-11, enero-junio 2018.
ISSN 1665-0654, e-ISSN en trámite, Instituto Politécnico Nacional México

Análisis de software aeroespacial mediante un algoritmo de planificación modular aviónico

Verónica **Quintero-Rosas**
Claudia **Martínez-Castillo**
Heber **Hernández-Tabares**
Mario **Camarillo-Ramos**
Gilberto **García-Gómez**
Francisco **Ibáñez-Salas**
José **Juárez-Viveros**

Tecnológico Nacional de México
Instituto Tecnológico de Mexicali
Departamento de Sistemas y Computación
Av. Tecnológico s/n, 21254
Mexicali B.C.
MÉXICO.

Tel. (686) 2273518
correo electrónico (email):
veronicaquintero@itmexicali.edu.mx

Recibido 31-07-2017, aceptado 13-10-2017.

Resumen

Todo lo referente a aeronáutica desde su fabricación, hasta su implementación debe tener un constante monitoreo y análisis de seguridad. La respuesta a un fallo, no puede ser improvisado. La seguridad en el sector industrial aeroespacial es un punto crítico e importante a tratar, respecto a su infraestructura y tecnología. Actualmente los diseños de software y hardware han evolucionado respecto a modelos anteriores, donde los controladores eran exclusivamente analógicos y los modelos se diseñaban con procesos independientes en forma de módulos aislados, pero interconectados entre sí. Este diseño punto a punto permitía que si un proceso o módulo fallara, este no afectaría al sistema en conjunto. Se divide en cuatro módulos: control de motor, vuelo, combustible y sistema de aire. Los avances tecnológicos actuales han contribuido a nuevos sensores, nuevos protocolos de comunicación, mejor procesamiento y ahora con los sistemas ciberfísicos, los circuitos y software de redundancia e inclusive software múltiple han dado lugar a nuevas certificaciones y normas de índole federal e internacional. En este documento analizaremos las diferentes problemáticas y retos a los que se enfrenta la seguridad respecto a software con tecnología ciberfísica y verificaremos un algoritmo propuesto, respecto a algoritmos utilizados actualmente. Simularemos el algoritmo dentro de un sistema empotrado y encriptado utilizando el algoritmo de Huffman y daremos a conocer los resultados del experimento, sus retos y conclusiones.

Palabras clave: industria aeroespacial, seguridad en software, algoritmos.

Abstract
(Aerospace Analysis Software Using an Algorithm of Avionic Modular Planning)

Everything related to aeronautics from its manufacture, until its implementation must have a constant monitoring and security analysis. The answer to a fail, cannot be improvised. Security in the aerospace industry is a critical and important point to argue, with respect to infrastructure and technology. Currently, software and hardware designs have evolved from earlier models, where controllers were exclusively analog and the models were designed with independent processes in the form of isolated but interconnected modules. This point-to-point design allowed that if a process or module failed this would not affect the system as a whole. It is divided into four modules: engine control, flight, fuel and air system. Current technological advances have contributed to new sensors, new communication protocols, improved processing and now with cyberphysical systems, circuits and redundancy software and even multiple software have given rise to new certifications and standards federal and internationals. In this paper, we will analyze the different problems and challenges that security faces regarding software with cyberphysical technology and we will verify a proposed algorithm, regarding algorithms currently used. We will simulate the algorithm inside an embedded and encrypted system using the Huffman algorithm and we will give the results of the experiment, its challenges and conclusions.

Index terms: aerospace industry, software security, algorithms.

1. Introducción

Los sistemas de software aeronáutico para gestión de vuelo, ruta de vuelo, cálculo de distancias, tiempo, combustible y curso fijado, se definen y rastrean en tiempo real [1]. Todo

esto se lleva a cabo mediante algoritmos e interconexiones de comunicación, de sensores de rumbo; los cuales detectan al magnetismo del norte de la tierra, sin olvidar los sensores de altitud, velocidad, temperatura, etc. Esta integración de componentes y software se define como Integración Modular Aviónica (IMA) [2], el cual define una red distribuida de componentes en tiempo real.

Las normas Federales de Procesamiento de Información (FIPS) [3], son normas desarrolladas por el gobierno federal de los Estados Unidos para el uso de sistemas informáticos de agencias gubernamentales no militares y contratistas.

Estas normas establecen requisitos para garantizar la seguridad informática y algunas especificaciones son modificaciones de los estándares de ANSI (Instituto Nacional de Estándares Americanos), IEEE (Instituto de Ingenieros Eléctricos y Electrónicos), etc. [4]. Las consideraciones de seguridad de los sistemas de software aeroespacial [5] radican en la compatibilidad de: hardware-software, software-software, aeronave-software. Dando lugar a técnicas para detección de daños, efectos de carga, errores de software, entre otros. Actualmente existen métodos de encriptado [6] que permiten aumentar la seguridad en la comunicación del software y de hardware.

En el software, un mensaje es utilizado mediante la codificación del contenido; de tal forma que solo pueda ser decodificado únicamente a quien fue asignado; utiliza básicamente claves en forma de tokens.

El estándar más utilizado en sistemas aeroespaciales para encriptación es AES (*Avance Encryption Standard*) [7], el cual es un algoritmo clasificado por la Agencia de Seguridad Nacional de los Estados Unidos (NSA) [8]. El algoritmo de AES está basado en una transformación lineal de permutaciones y codificación simétrica. En este artículo analizaremos este cifrado respecto al algoritmo de adaptación de Huffman [9] utilizando sistemas ciberfisicos [10]. Ya que los sistemas aeronáuticos abarcan desde piloto automático, instrumentos de navegación, controles de motores, parámetros de vuelo, sistemas de navegación (GPS, vor), comunicaciones (control de tránsito aéreo, sistemas de posicionamiento global) entre otros; estos deben estar interconectados respecto a la comunicación de software pero de una manera codificada y segura.

En este artículo verificaremos el uso de la codificación Huffman mediante un token centinela (ciberfísico), el cual supervisa fallas de hardware-software y da informe de estado. Ya que la seguridad de las interfaces del sistema al comunicar hardware y software pueden causar fallas en la seguridad de vuelo del avión.

Debido a la línea de comunicación de circuitos electrónicos, software de redundancia y software múltiple [11], nace aviónica modular integrada [12]; la cual es muy importante para la comprensión de la problemática en la industria aeroespacial y se explica en la siguiente sección.

2. Aviónica modular integrada (IMA)

La aviónica modular integrada [13] es una red distribuida en tiempo real, instalada abordo de una aeronave. No es propiamente un sistema operativo, ya que son sistemas independientes. La red consta de varios módulos de procesamiento y aplicaciones. Se encuentra en el estándar ARINC 65 [14]. Se basa en la arquitectura de capas, la cual permite aislar aplicaciones del hardware/software. Tiene mecanismos de protección para permitir que los recursos puedan ser compartidos por diferentes aplicaciones con distintos niveles de importancia o jerarquía. También debe permitir que las aplicaciones se inserten o extraigan del sistema sin que causen ningún impacto en el mismo. A estas técnicas, se denomina particiones. En la programación y configuración de software ciberfisico [15] se debe integrar los módulos de sensores inalámbricos [16] y la integración física entre redes, módulos y dispositivos de entrada y salida. Un ejemplo de ello son las redes Gate Link, donde se puede recibir y transmitir información desde y hacia la aeronave.

En México [17], los productos y servicios aeroespaciales incluyen componentes del sistema de propulsión, aeroestructuras, componentes de los sistemas de aterrizaje, sistemas de frenado, partes mecanizadas de precisión, piezas de moldeo por inyección de plástico, partes eléctricas y electrónicas, diseño y servicios de ingeniería e interiores de aviones, entre otras; pero no incluye el diseño de software. Hasta el día de hoy la mayoría de la industria aeroespacial no estaba conectada con el mundo exterior, por lo que la seguridad era más robusta. Con las nuevas tecnologías ahora es necesario proteger tanto los instrumentos, el control, programas, etc.; además de toda la plataforma del sistema del software. Es importante resaltar que la nueva industria aeroespacial, es ahora una e-industria automatizada, donde los sistemas ciberfísicos y las aplicaciones basadas en la nube [18] son una realidad. La caracterización se basa en sensores, un sistema de control y un sistema de comunicación interactuando con procesos, protegiendo el buen funcionamiento todo el sistema de la aeronave. En la aviónica modular integrada para un sistema ciberfísico, se debe tener en consideración la reutilización de código, así como su portabilidad y su eficacia en las actualizaciones.

3. Trabajo relacionado

GMV [19] desarrolla sistemas aeronáuticos de aproximación y aterrizaje basados en la navegación por satélite. Básicamente es un proveedor de importantes fabricantes aeronáuticos presta servicios de ingeniería y desarrolla avanzados sistemas y programas para el sector aeronáutico. Tiene servicios en las áreas de ingeniería de sistemas, software para aviónica y seguridad crítica, simuladores, bancos de prueba, entre otras.

Aunque GMS desarrolla toda una infraestructura de diseño, existe software específicamente utilizado en la industria de ingeniería de sistemas aeronáuticos, como PLM [20] de Siemens, el cual incluye procesos de desarrollo, ingeniería estructural, mecánica, de sistemas y ensayos ambientales. Es utilizada también en la industria automotriz, energética y marítima. Este sistema analiza el software de ingeniería de materiales compuestos, simula productividad que alertan sobre los posibles problemas o fallos.

Otra empresa importante desarrolladora de software es AERTEC [21], donde diseñan sistemas de software aeronáuticos, así como sistemas de navegación y control, tomando en consideración la operación en condiciones extremas.

Todos estos sistemas son basados en la protección de red a datos de acceso externo no autorizado, y de software cargable en campo, es decir software y base de datos que se puedan descargar sin retirar del equipo de la aeronave.

4. Normas de software en el sector aeroespacial

O-178B [22]. Son normas y certificaciones para software en sistemas y equipos aeroespaciales. Básicamente es una guía de la seguridad del software y es un estándar para el desarrollo de sistemas de software de aviónica.

DO-326A [23]. Esta norma estipula los procesos de seguridad, el proceso de evaluación (SAE ARP 4761) y el proceso de ingeniería del sistema (SAE ARP 4754A).

DO-178B [24]. Certificación para evidenciar el cumplimiento de requisitos de seguridad. Asigna planes de seguridad de software IEEE STD-1228-1994 y define tareas de análisis de seguridad de software en pasos secuenciales (análisis de requisitos, análisis de diseño de nivel superior, análisis de diseño detallado, análisis de nivel de código, análisis de pruebas y análisis de cambios). Cualquier software que ordene, controle y monitoree las funciones críticas de seguridad debe estar certificado bajo esta norma.

FAA (Administración Federal de Aviación) [25]. Aplica DO-178B como guía para determinar si el software funcionará confiablemente en un entorno aeroespacial. En los Estados Unidos, el DO-178B se establece en el título 14: Aeronáutica y Espacio del Código de Reglamentos Federales.

IDAL: Determina el nivel de garantía del software, es decir determina a partir de procesos de evaluación y análisis de peligros en caso de fallo en el sistema. Las condiciones de falla se clasifican por sus efectos en la aeronave, la tripulación y los pasajeros en [23] [25]:

a) Catastrófico. La falla puede causar un accidente fatal. Causa pérdida de la función necesaria para volar y aterrizar con seguridad.
b) Peligroso. Reduce la capacidad de la tripulación para operar la aeronave debido a sufrimiento físico, o lesiones graves o mortales entre los pasajeros.
c) Mayor. Tiene un impacto menor que un fallo peligroso, puede causar molestias físicas de presión, mareo sin llegar a causar lesiones.
d) Menor. Fallo notable, puede causar inconvenientes como cambios de plan de vuelo.
e) Sin efecto. El fallo no tiene impacto en la operación de la aeronave.

ARINC-653[14]. Es un estándar que define el entorno de ejecución de aplicaciones, puede utilizarse siempre que múltiples aplicaciones necesiten compartir un único procesador/memoria, con el fin de garantizar un acceso exclusivo de los recursos físicos y temporales a las aplicaciones mientras que comparte recursos; y garantiza que una aplicación no puede afectar a otra en caso de fallo de la aplicación. En el caso de los sistemas empotrados, ARINC 653 define un esquema de particionado, define critica la memoria caché, ya que es limitada.

5. Algoritmos de software aeroespacial

Los sistemas aeroespaciales [26] son cada vez más dependientes del software, pero aún tienen desafíos basados en aplicaciones en tiempo real y algoritmos que deben ser modificados para las nuevas arquitecturas de comunicación; ya que aún son utilizados sistemas analógicos además de los sistemas de redundancia digitales.

Algoritmos Vand [27] (búsqueda en profundidad). Consisten en encontrar la mejor solución posible, en un tiempo determinado. Si en la búsqueda a una solución se encuentra una alternativa incorrecta, la búsqueda retrocede hasta el paso

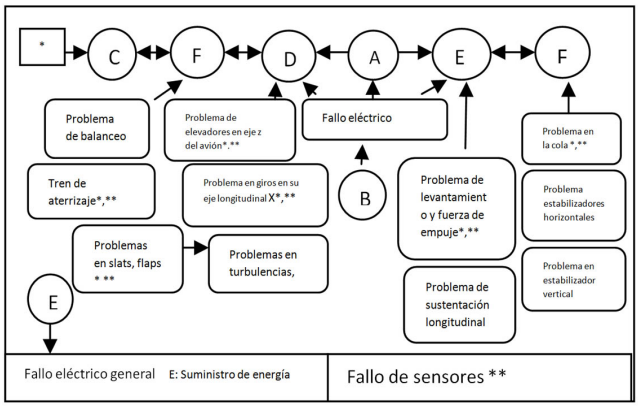

Fig. 1. Monitoreo general del sistema para estabilizar la nave.

anterior y toma la siguiente alternativa; es decir, utiliza la recursividad tal que llama al procedimiento para cada uno de los nuevos estados. La diferencia con los algoritmos de recursividad, es que diseña funciones de corte para no generar estados que no den solución o den una solución peor de la que se tiene. Es similar a los algoritmos de árbol padre-hijo, pero este falla cuando la búsqueda ya no tiene más alternativas. Así se crea un árbol subyacente, cada nodo es un estado de la solución parcial o final. Este algoritmo de profundidad se aplica generalmente en fallos de la aeronave, para reducir un impacto eminente de la nave. Un ejemplo de ello es la figura 1, donde el sistema marcara un fallo general y este inicia un protocolo de chequeo en todo el sistema, para estabilizar la aeronave con las condiciones posibles. Las condiciones generales a revisar son: A: Combustible, B: Comunicaciones, C: Oxígeno, D: Turbina 1, E: Turbina 2, F: Fuselaje, G: Energía.

Esta configuración cambia para cada tipo y modelo de avión, por ejemplo el tipo de motor (turbo jet, turbo hélice, etc.). Esto afecta a las fuerzas que actúan en la aeronave tanto en la sustentación (fuerza que permite mantenerse en el aire), el peso (fuerza de gravedad), la resistencia (fuerza de oposición), el empuje (fuerza del motor opuesta a la resistencia).

Aunque existen códigos de funcionalidad universal [28] como los problemas de rotación de la aeronave donde los elevadores (estabilizadores horizontales de la cola) permiten rotar en su eje lateral *y*, de llegar a fallar habría un serio problema para ascender y descender en el aire. Este problema generalmente afecta también a la dirección del avión (eje vertical *z*) el cual da rumbo a la dirección trazada. Algoritmos de reemplazo de páginas. Este algoritmo [29] da referencia a páginas, modificaciones, y está ligado a estampillas de tiempo para hacer dichas referencias. Encuentra páginas de reciente uso y reduce el consumo de energía.

6. Lógica de intercambio de información entre aplicaciones

La lógica de intercambio de información entre aplicaciones en un sistema aeronáutico [30] se da mediante una interfaz de software que sincroniza, enlaza y da seguridad a una aplicación; para interactuar con otras aplicaciones incluyendo hardware y diferentes tipos de sistemas operativos. Este intercambio de información entre aplicaciones ayuda en la complejidad de la comunicación en sistemas distribuidos y heterogéneos. Aún se encuentra en desarrollo ya que los proveedores en la línea de software espacial han visualizado que el desarrollo de código abierto es muy eficaz en la portabilidad y la lógica del intercambio de información entre las aplicaciones. Las investigaciones financiadas por el Departamento de Defensa de Estados Unidos, ha visualizado la importancia de la lógica de intercambio de información con los sistemas en tiempo real para simplificar el desarrollo de aplicaciones; ya que a medida que las redes de sistemas aeroespaciales crecen, la diversidad de la funcionalidad integrada también aumenta.

Este intercambio de información debe darse en diferentes capas y subcapas, siendo las más utilizadas:

1. Trazo de ruta
2. Piloto automático
3. Sensores
4. Señales de navegación
5. Comunicación

Recordando que los criterios de seguridad en software están definidos por: la arquitectura de redundancia en hardware/software, flujo de datos, rendimiento, velocidad de respuesta y tolerancia a fallos dentro de sistemas embebidos propios de la aeronave. Actualmente este intercambio entre aplicaciones ya está regulado mediante la RTCA [31] (Comisión Radiotécnica para la Aeronáutica); la cual realiza recomendaciones y son base en la FAA (Administración Federal de Aviación de los Estados Unidos). Dichas recomendaciones certifica el software para aeronaves. A pesar de la utilización digital, actualmente se conserva en los sistemas aeronáuticos una copia de seguridad análoga en los sistemas críticos, por lo que la redundancia tanto en hardware como en software sigue permaneciendo. Cada línea o ruta redundante se denomina canal. Estos canales incluyen sensores, cableado y convertidores análogo-digital, digital-analógico. Se utilizan algoritmos de gestión para determinar si una entrada o res-

puesta es redundante, o si ya está siendo procesada. De tal forma que existen múltiples canales procesando información simultáneamente, y estos deben tener una sincronización para dar respuesta. No se debe olvidar que toda reconfiguración de software debe estar presente en casos de fallo ya que esto preserva, aísla y da seguridad. Los sistemas aeronáuticos integran miles de componentes y conexiones, lo cual aumenta la complejidad de configuración y esto a su vez aumenta la posibilidad de fallos.

En la aeronáutica militar es utilizado el llamado Guardas de Dominio Cruzado (CDG) [32] , el cual permite que la información pueda ser transferida entre dos o más dominios de seguridad diferentes. Incluso poder comunicar sensores o hardware con el sistema mediante esta interfaz, es decir, que una red aislada intercambie datos con otra red sin perder la seguridad que conlleva el conectarse con una red distinta. Esto permite la transferencia de información entre dominios o niveles de clasificación incompatibles. Otra técnica para resguardar el intercambio de información entre aplicaciones o sistema en general, es mediante el guardado periódico del estado de la aplicación; haciendo recuperaciones de la información constantemente.

Existen muchos retos en el intercambio de información, entre estos desafíos se encuentra el diseño de aplicaciones para monitorear la caja negra de una aeronave. Actualmente no existe una revisión constante de la caja negra [33] (realmente es color naranja), y se dan casos donde en un fatal accidente al revisar la caja negra esta se encuentra sin información. La caja negra es un módulo muy importante donde se graban las acciones tomadas por los pilotos, instrumentos e inclusive audio. Existe una caja negra de acceso rápido, la cual se encuentra generalmente al frente de la aeronave pero esta no sobrevive en un accidente por su posición. Esta caja de acceso rápido puede dar datos al igual que la caja negra principal, pero con datos restringidos. Actualmente los diseños de última generación podrán enviar los datos de la caja negra principal a una nube y podrán ser leídos mediante una aplicación constantemente; dando disponibilidad y fiabilidad e inclusive video, sin necesidad de que la caja negra se envié a Washington para ser analizada por especialistas y solo en casos de accidentes fatales. Es en esta etapa donde los sistemas ciberfisicos tienen un papel muy importante, ya que la tecnología ciberfísica [34] demuestra una nueva forma de comunicación y colaboración en los distintos equipos de software y hardware en tiempo real, dando interconexión de sensores, actuadores, controladores, comunicaciones, etc. Es importante resaltar la técnica de los sistemas ciberfísicos, la cual es por realimentación de estados y a su vez compara los resultados con un controlador de planificaciones; es decir un controlador x, con retroalimentación de salida o estados, el cual garantiza que el sistema en lazo cerrado se encuentre en el estado próximo requerido. Para ello se utilizan diferentes componentes físicos (sensores) tanto en software como hardware, de forma integrada e interactuando para detectar y controlar en tiempo real el estado del planificador de acciones de vuelo, monitoreando, controlando, supervisando y dando reportes de todo el sistema. Es una aplicación embebida ubicua [35] en el cual se comparte hardware, software, sensores, electrónica para comunicar, diagnosticar, almacenar e interconectar multifactorialmente todo el proceso de manera sistemática. Es decir, trabajando de manera modular e independiente todo el sistema aéreo, pero definido en una arquitectura ciberfísica que analiza todo el sistema como un ecosistema complejo.

7. Algoritmo propuesto

La caracterización de los sistemas ciberfísicos se basan en sensores, un sistema de control y un sistema de comunicación interactuando con procesos propios de la aeronave; para el buen funcionamiento todo el sistema. Existen programas realizados en lenguajes de programación Ada, Prolog, Joval, C, entre otras. Todas ellas son utilizadas en diferentes módulos (comunicación, energía, combustible, etc.). En estos módulos independientes, un sistema ciberfisico debe ser capaz de monitorear y controlar dichos módulos; pero además debe verificar el sistema de gestión de vuelo, los enlaces de datos del piloto con el controlador de comunicaciones, controles automáticos del sistema de guía de vuelo además de tener algoritmos de verificación por redundancia cíclica. Existen estándares lo cuales no son propiamente algoritmos, pero certifican a los algoritmos utilizados en la aviónica como lo es el Estándar de Cifrado Avanzado (AES) [36]; el cual muestra las especificación para el cifrado de los datos, establecidos por el Instituto Nacional de Estándares y Tecnología de los Estados Unidos (NIST) [37]. Estos algoritmos utilizan la misma clave para codificar y decodificar, ejemplo de ellos son los algoritmos vistos.

El algoritmo ciberfisico propuesto se basa en un token entrenador y un token centinela. El token entrenador será quien simule antes de enviar una señal de salida a las posibles combinaciones de respuesta, para emitir la mejor respuesta. El token centinela será quien verifique dicha respuesta respecto al historial. El historial se encuentra en una nube la cual tendrá rutas de vuelo anteriores, registros de datos de instrumentos, datos generales del vuelo, etc. El algoritmo tendrá una etapa de entrenamiento después de 15 rutas del punto A al B. En caso de que se trate de un avión comercial y cambie su ruta de vuelo, el algoritmo podrá verificar la base de datos de la nube respecto a su nuevo vuelo (si el avión es del mismo modelo, caso de los

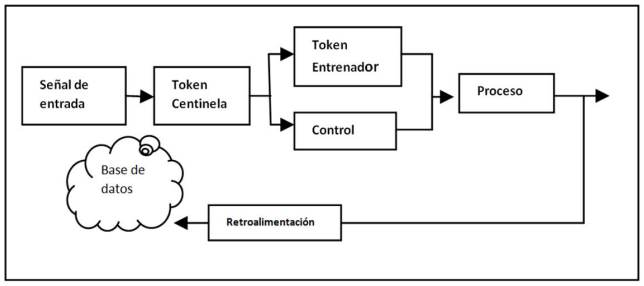

Fig. 2. Algoritmo ciberfisico propuesto.

aviones comerciales) y este podrá utilizar dicho token de entrenamiento; como lo muestra la figura 2.

El encriptado de la información será mediante la codificación de huffman para hacer dicha comunicación más eficiente, rápida y segura. Esta codificación de Huffman esta descrita en un método de construcción mínimo de redundancia, por esa razón es tan eficiente y comprimirá la información para él envió a la nube. Los parámetros de nuestro caso de estudio es el siguiente, aplicado al proceso de tren de aterrizaje (véase figura 3):

Señal de entrada (S0), bajar tren de aterrizaje;

Token entrenador (Ǝ), verificación de velocidad vertical de aterrizaje en vuelos anteriores, de misma ruta;

Control (C), verificación en tiempos respecto a sensor de neumáticos, amortiguadores;

Token centinela (Б, verifica en nube, tiempo de vuelo y tiempo promedio de acción del tren de aterrizaje; en mismas rutas, de vuelos anteriores;

Proceso, acción;

Retroalimentación (R): registro en la nube de todo el proceso y asignación de estampillas de tiempo a dicho registro.

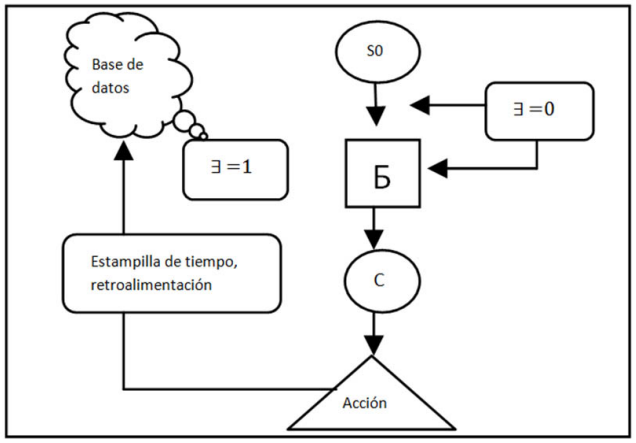

Fig. 3. Caso de estudio aplicado al tren de aterrizaje.

Tabla 1. Caso de estudio utilizando algoritmo ciberfísico

Acción del piloto	Token Centinela	Token Entrenador
Encendió del tren de aterrizaje	Registro en la nube $1039	LDAA #$80 STAA $1039
Selección del canal	Registro en la nube $1030	LDAA #$22 STAA $1030
Latencia Huffman	100 m/seg.	JSR Latencia
Resultado	Registro en la nube $1031-4	LDAA $1033
Limpiar registro		CLR $1039 CLR $1030

Es importante recordar que estos sistemas pueden estar dentro de un sistema embebido o muy especializado, siendo este una parte de un sistema más grande. Para su programación se utilizan micro controladores, los cuales están alojados en una placa del microprocesador. En nuestro caso de estudio, la codificación se simulo en un micro 68HC11 de Motorola [38], de la siguiente manera (véase tabla 1 y figura 4).

```
FF06-D7 Sistema
FF08-D9 Transferencia Huffman
FFDA-DB Entrada de reloj (sincronización)
FFDC-DD Sobre flujo (estampilla de
tiempo)
FFDE-DF Reloj final de sobre flujo
FFE0-E1 Salida de comparación a la nube
FFE1-E2 Salida de comparación a la nube
        .
        .
FFE0-E15 Salida de comparación a la nube
FFEA-EB Estampilla de tiempo 1
FFEC-ED Estampilla de tiempo 2
        .
        .
FFF0-F1 Control
FFF2-F3  /IRQ del proceso…
FFFE-FF /Reset
```

```
Tentrenador: Rmb1
LDS $0055   // inicializa stack
CLR $1008  //limpiar puerto
LDAA #$7E  //carga código Huffman
STAA $00EE //carga dirección en nube
LDD #IRQ
STD $00EF
Tcentinela: LDAB #$10
STAB $1009
IRQ: LDAA #$20
RTI
```

Fig. 4. Codificación en tarjeta 68HC11.

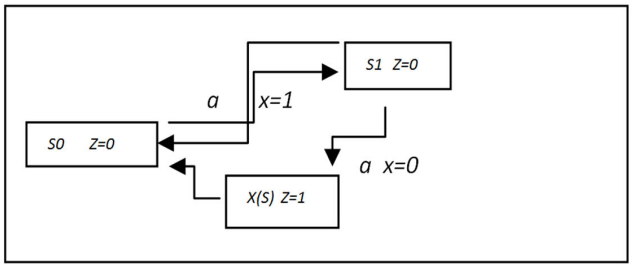

Fig. 5. Algoritmo de planificación.

Los algoritmos criptográficos [39] utilizados actualmente son basados en estructuras algebraicas de anillos de tipo no conmutativa; por lo que los procesadores depuran algoritmos pequeños que no afectan al sistema y esto los hace más seguros. Por tal razón este estudio está definido en un sistema planificador por lo que se utilizó un algoritmo autónomo de planificación (figura 5); donde toma una serie de acciones que resultan en un conjunto de tareas. Está definido por el escenario virtual S(0) en una distribución de estados S(0) = x (S). Dado el escenario actual S(1) donde a es la acción, z es la respuesta.

El pseudocódigo del planificador se muestra en la figura 6 y el pseudocódigo de Huffman en la figura 7.

El tiempo estimado (véase figura 8) para esta fase piloto en la tarjeta HC11 fue de 15 estados, con señal normalizada. En el caso de un sobre flujo en el proceso 11, 12, 26, 27, 28 se nota una descompensación en la fase de entrenamiento siendo el

```
BAUD RMD1 //Reserva la dirección en memoria
SCR1 RMD1
SCCR2 RMD1
SCDR RMD1
SCSR RMD1
ORG $B600 //Empieza
INICIO: LDAA #$2C
STAA SCCR2
LDAA #$30
STAA BAUD
LDAA #$7E
STAA $00C4
CICLO: CLR $0060
BRA CICLO
LDAB SCSR
SERIE 1: ANDB #$20
BEQ SERIE 1
LDAA SCDR
```

Fig. 6. Codificación del planificador.

```
HuffmanList lista=new HuffmanList ();
 for(int i=0;i<cadenalength();i++){
bitx=(byte)cadena.charAt(i);
cadena=Integer.toBinaryString(bitx & 0xFF);
if(cadena.length()<8)
        cadena=sello(cadena);
        if(i==(cadena.length()-1))
        // decodificación
                for(int i=0;i<Acodificar.length();i++)
                cadena=cadena+Acodificar.charAt(i);
                        cadena=tabla.LetraDe(cadena);
```

Fig. 7. Codificación de Huffman.

agente centinela más estable después de un sobre flujo o error de la información. Sin embargo su recuperación de proceso es alentadora (véase figura 9).

El trabajo en paralelo del algoritmo, respecto a los atributos de tiempo de respuesta y verificación, así como la organización mediante un planificador de tareas responde bien en 50 acciones.

Sin embargo, después de estas ejecuciones empieza una saturación de memoria que después de limpiarla no responde adecuadamente al planificador, dando solo respuesta a la acción inicial con el token centinela pero no verifica en la base de datos los casos anteriores.

Esto se resolvió agregando una tarjeta BeagleBone [40] dedicado a planificar las tareas, resolviendo en una sola señal de salida en que proceso continuaría (activar flaps, spoilers de las alas, etc.).

Señal		Control	Б	Proceso	Estampilla de T.
Activacion del tren de aterrizaje A1	.06 NanoTime	.0001 NT	.08 NT	Accion	.00002 NT
A2	.03 NT	.0001 NT	.03 NT	Accion	.00001 NT
A3	.03 NT	.0001 NT	.03 NT	Accion	.00001 NT
A4	.02 NT	.0001 NT	.04 NT	Accion	.00003 NT
A5	.03 NT	.0001 NT	.03 NT	Accion	.00001 NT
A6	.04 NT	.0001 NT	.04 NT	Accion	.00001 NT
A7	.04 NT	.0001 NT	.04 NT	Accion	.00002 NT
A8	.05 NT	.0001 NT	.04 NT	Accion	.00001 NT
A9	.05 NT	.0001 NT	.04 NT	Accion	.00001 NT
A10	.05 NT	.0001 NT	.04 NT	Accion	.00001 NT
A11	.05 NT	.0001 NT	.04 NT	Accion	.00002 NT
A12	.05 NT	.0001 NT	.04 NT	Accion	.00001 NT
A13	.05 NT	.0001 NT	.04 NT	Accion	.00001 NT
A14	.05 NT	.0001 NT	.04 NT	Accion	.00001 NT
A15	.05 NT	.0001 NT	.04 NT	Accion	.00001 NT

Fig. 8. Tabla de tiempo total por proceso.

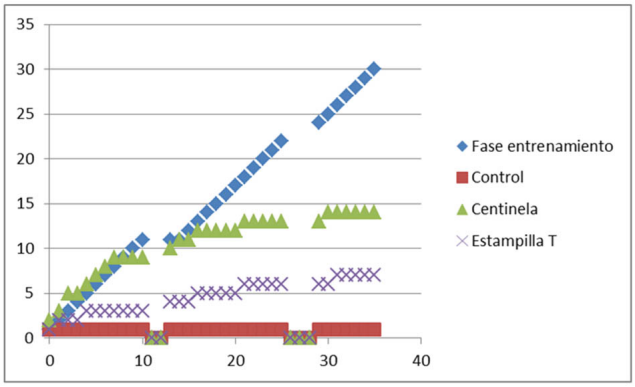

Fig. 9. Comportamiento en caso de error o sobre flujo en la información.

8. Conclusiones

Es necesario hacer un estudio más profundo respecto a nuestro caso de estudio, ya que el algoritmo propuesto solo fue experimentado bajo circunstancias de dos procesos de planificación en un esquema de codificación huffman. El poder simular en la tarjeta HC11 el proceso de error de un sobre flujo o error de comunicación nos permitió visualizar el comportamiento del algoritmo, y su forma de recuperación al sistema dando seguridad y funcionamiento. Sin embargo, necesitamos hacer más pruebas a profundidad para verificar porque después de cincuenta procesos el token de la nube se satura y no verifica en la nube los datos registrados anteriormente. Hasta el día de hoy el poner una tarjeta dedicada al planificador de tareas resolvió el problema, pero debemos replicar el experimento sin la utilización de la tarjeta Beagle Bone para tener un sistema de redundancia a partir de que Beagle Bone pudiera fallar también. La seguridad y la compatibilidad son muy importantes en este sistema, ya que depende del correcto funcionamiento del sistema de control utilizando técnicas intrusivas. Utiliza encapsulamiento y fragmentos en el nivel de aplicación (mensaje), los cuales se consideran segmentos (trasporte) y tramas en el nivel enlace. Como trabajo futuro la interfaz de simulación entre distintos dispositivos, sensores, software de doble seguridad en simulación siempre se comunica con la nube, sin embargo en la seguridad aeroportuaria esto no es posible, ya que no está permitido ningún uso de dispositivos de comunicación.

La mayoría de la industria aeroespacial no estaba conectada con el mundo exterior, por lo que la seguridad era más robusta. Con las nuevas tecnologías ahora se tiene que proteger tanto los instrumentos, el control, los programas; además de toda la plataforma de software y hardware. Se deben tener mecanismos de evaluación de riesgos, auto calibración de software y una planificación de seguridad. Hace falta profundizar y enlazar estos algoritmos de seguridad a los procesos anteriores y posteriores para que esto sea realmente un sistema ubicuo ciberfisico [41]. El control automático e inteligente de todos los procesos con conexión directa a internet puede perturbar el control y funcionamiento general de la aeronave. Los candados de seguridad para el software que comunica sensores, planificadores, base de datos no deben trabajar individualmente, si no como un universo de candados de seguridad con mecanismos multifuncionales que protejan tanto a software como hardware.

Referencias

[1] J. Cooling, *Software Engineering for Real-Time Systems*, USA: Addison-Wesley, 2002. ISBN-13: 978-0201596205

[2] R. Wolfig, *A Distributed Platform for Integrated Modular Avionics: Insights on Future Avionics Systems and their Certification Requirements*. Alemania, Verlag, 2008. ISBN-13: 978-3838100494

[3] National Institute of Standards and Technology, *Federal Information Processing Standards Publications*. ISBN-13: 978-1547148240

[4] Standards Information Network. *IEEE Standards Dictionary: Glossary of Terms and Definitions*. ISBN-13: 978-0738157399

[5] C. Anderson & M. Dorfman, *Aerospace Software Engineering: A Collection of Concepts*, 1991. ISBN-13: 978-1563470059

[6] L. Rierson, *Developing Safety-Critical Software: A Practical Guide for Aviation Software and DO-178C Compliance*, USA: CRC Press, 2013. ISBN-13: 978-1439813683

[7] C. Cid, S. Murphy, & M. Robshaw, *Algebraic Aspects of the Advanced Encryption Standard*. USA: Springer, 2006. ISBN-13: 978-0387243634

[8] L. K. Johnson, *National Security Intelligence*, USA: Polity Press, 2012. ISBN-13: 978-1509513055

[9] K. Sayood, *Introduction to Data Compression*. USA: Elsevier, 2012. ISBN-13: 978-0128094747

[10] D. B. Rawat, J. Rodrigues, & I. Stojmenovic, *Cyber-Physical Systems: From Theory to Practice*. USA: CRC Press, 2015. ISBN-13: 978-1482263329

[11] I. Troch, *Simulation of Control Systems: With Special Emphasis on Modelling and Redundancy*. USA: Elsevier, 1978. ISBN-13: 978-0444851994

[12] Naval Postgraduate School, *Software-Defined Avionics and Mission Systems in Future Vertical Lift Aircraft*. USA: Penny Hill Press, 2016. ISBN-13: 978-1523265145

[13] Metrology and testing czeh office for standars, *CSN EN 4660-005 - Aerospace series - Modular and Open*

Avionics Architectures - Part 005: Software. ASIN: B01FXCO6SG

[14] Hephaestus Books, *Aviation standards, including: inter-range instrumentation, ARINC 429, DO-242A, AS9100, ARINC 661, ARINC 708, AS9000, ARINC 653, allied standar*, 2011. ASIN: B007S78CHQ

[15] R. Rajkumar, I. Lee, & L. Sha, & J. Stankovic, *Cyber-Physical Systems (SEI Series in Software Engineering)*, Proceedings of the 47th Design Automation Conference, Anaheim, California, June 13-18, 2010, pp. 731-736.

[16] J. Garbarino. *Protocolos para Redes Inalámbricas de Sensores,* Tesis de Ingeniería en Informática, Buenos Aires, Argentina, 2011.

[17] Federación mexicana de la industria aeroespacial, 2017. Disponible en: http://www.femia.com.mx/

[18] G. Toraldo, *OpenNebula 3 Cloud Computing*. USA: PACKT, 2012. ISBN-13: 978-1849517461

[19] *Sistemas Aeronauticos*. Disponible en http:// www.gmv.com/pt/Sectores/aeronautica/

[20] *Industria de ingeniería de sistemas aeronáuticos*, 2017. Disponible en : https://www.plm.automation. siemens.com/es_mx/aerospace-defense/space-systems/

[21] *Desarrollo de software aeroespacial*, 2017. Disponible en www.aertecsolutions.com/ingenieria-de-sistemas/

[22] M. Klamert, *Services Liberalization in the EU and the WTO: Concepts, Standards and Regulatory Approaches*.UK: Cambridge University Press, 2014. ISBN-13: 978-1107034594

[23] American National Standards Institute, 2017. Disponible en https://www.ansi.org/

[24] V. Hilderman, & T. Baghai, *Avionics Certification: A Complete Guide to DO-178 (Software), DO-254 (Hardware)*. USA: Avionics Communications, 2007. ISBN-13: 978-1885544254

[25] Administracion Federal de Aviacion, 2017. Disponible en https://www.faa.gov/

[26] P. Belobaba, A. Odoni, & C. Barnhart, *The Global Airline Industry (Aerospace Series)*. USA: Wiley, 2015. ISBN-13: 978-1118881170

[27] R. Sedgewick, *Algorithms*. USA: Addison-Wesley, 2011. ISBN-13: 978-0321573513

[28] J. Prior, *Aviones y su funcionamiento*. USA: Time, 2005. ISBN-13: 978-0743900393

[29] S. Cardona, S. Jaramillo, & J. Triviño, *Análisis de Algoritmos para Ingeniería de Sistemas y Computación*. México: Elizcom, 2012. ASIN: B00GG6AKF2

[30] D. Kritzinge, *Aircraft System Safety: Military and Civil Aeronautical Applications*. USA: Elsevier, 2006. ASIN: B00M3Z1I9C

[31] Comisión Radiotécnica para la Aeronáutica, 2017. Disponible en : https://www.rtca.org/

[32] Cross-domain guards (CDG), 2017. Disponible en : https://www.deep-secure.com/

[33] M. Soler, *Fundamentals of aerospace engineering: An introductory course to aeronautical engineering*. USA: Createspace Independent Pub, 2017. ISBN-13: 978-1493727759

[34] S. C. Suh, U. J. Tanik, J. N. Carbone, & A. Eroglu. *Applied Cyber-Physical Systems*. USA: Springer, 2014. ISBN-13: 978-1461473350

[35] D. Molloy, *Exploring BeagleBone: Tools and Techniques for Building with Embedded Linux*. USA: Wiley, 2015. ISBN-13: 978-1118935125

[36] J. Daemen, & V. Rijmen, *The Design of Rijndael: AES - The Advanced Encryption Standard*. USA: Springer 2002. ISBN-13: 978-3642076466

[37] Information Systems Audit and Control Association, *Implementing the NIST Cybersecurity Framework*. USA: ISACA, 2014. ISBN-13: 978-1604203578

[38] J. Park, *Practical Embedded Controllers: Design and Troubleshooting with the Motorola 68HC11*. USA: Elsevier, 2003. ASIN: B001BOT7TE

[39] B. L. Summers, *Software Engineering Reviews and Audits*. USA: CRC Press, 2011. ISBN-13: 978-1439851456

[40] B. McLaughlin, *The BeagleBone Black Primer*. USA: Pearson, 2015. ISBN-13: 978-0789753861

[41] F. Maciá, *Computación ubicua*. España: Universitat d' alacant (publicacions), 2007. ISBN-13: 978-8479089115