Marco para la Aplicación de Criptoanálisis

Antonio Castro Lechtaler[1], Marcelo Cipriano[1], Francisco Javier Díaz[2], Edith García[1], Julio Liporace[1], Ariel Maiorano[1,2], Eduardo Malvacio[1]

[1] Facultad de Ingeniería del Ejército (FIE), Universidad de la Defensa Nacional (UNDEF)
{acastro,marcelocipriano,egarcia,jliporace,
emalvacio,maiorano}@fie.undef.edu.ar,
[2] Facultad de Informática (FI), Universidad Nacional de la Plata (UNLP)
{ariel.maiorano,javierd}@info.unlp.edu.ar

Resumen. El objetivo del presente artículo es presentar y describir un proyecto de software libre que implementa un marco o *framework* para la aplicación de diferentes metodologías, técnicas o mecanismos de criptoanálisis, conjuntamente con análisis auxiliares o complementarios. La herramienta permite aplicar diferentes técnicas sobre archivos de "muestras" manejados por el sistema. Permite a su vez la revisión comparativa de resultados. El código fuente ha sido publicado en Github.com.

Palabras clave: seguridad informática, criptografía, criptoanálisis, herramientas de criptoanálisis.

1 Introducción

Como fuera adelantado en el resumen, este trabajo presenta un proyecto de software libre y código abierto, desarrollado por el GICSI (Grupo de Investigación en Criptografía y Seguridad Informática) de la FIE (Facultad de Ingeniería del Ejército), para la implementación de un marco o *framework*. Se referirá al sistema como "Marco para la Aplicación de Criptoanálisis", o por sus siglas, MAC. Este sistema permitirá aplicar diferentes técnicas o mecanismos de criptoanálisis, conjuntamente con análisis auxiliares o complementarios, como por ejemplo análisis estadísticos sobre secuencias binarias. La herramienta permite aplicar estos análisis sobre archivos que llamaremos "muestras", también manejados por el sistema. Implementa a su vez la funcionalidad necesaria para la revisión comparativa de resultados.

Continuando en línea con otras publicaciones de software libre y de código abierto realizadas por el GICSI, relacionados a la seguridad informática en general y a la criptografía en particular [1,2]; la primer versión de este proyecto, MAC, que al momento permite la aplicación de análisis estadísticos y la comparación de los resultados obtenidos para diferentes muestras- ya se encuentra publicada en su totalidad y disponible sin restricciones en la plataforma de alojamiento de proyectos de software libre Github.com [3], bajo la Licencia Pública General de GNU versión 3, o GPLv3 por sus siglas en inglés.

2 Presentación del proyecto

2.1 Contexto

Este es un proyecto académico de desarrollo de software libre y de código abierto del Grupo de Investigación en Criptografía y Seguridad Informática (GICSI), de la Facultad de Ingeniería del Ejército (FIE), de la Universidad de la Defensa Nacional (UNDEF); forma parte del proyecto "Herramientas para la Evaluación de Algoritmos Criptográficos" (HEAC), aprobado en el Programa de Acreditación y Financiamiento de Proyectos de Investigación (UNDEFI), de la UNDEF.

A su vez, el proyecto se encuentra enmarcado en el trabajo realizado por miembros del GICSI doctorandos en el área "Redes y comunicaciones", sub-área "Criptografía", del Doctorado en Ciencias Informáticas de la Facultad de Informática (FI), de la Universidad Nacional de La Plata (UNLP).

Como fuera adelantado en la presentación de las líneas de investigación relativas al proyecto "Herramientas para la Evaluación de Algoritmos Criptográficos" (HEAC)[4], los objetivos incluyen el diseño y desarrollo de una herramienta que permita llevar adelante estos análisis y la realización de pruebas que permitan el estudio de diferentes propiedades de esquemas criptográficos de manera ágil y eficiente. Como parte de las primeras instancias o etapas en la implementación del proyecto, esta herramienta permite al momento el análisis estadístico de secuencias binarias para aplicar en algoritmos de cifrado de flujo (*stream ciphers*) y generadores de números pseudaleatorios (*pseudo random numbers generators*). Tales secuencias pueden ser generadas además por LFSRs (*Linear Feedback Shift Registers*), NLFSRs (*Non-Linear Feedback Shift Registers*), CCGs (*Clock Controlled Generators*), entre otros esquemas y mecanismos. Se espera diseñar y desarrollar un software abierto que permita la evaluación de las propiedades criptográficas y de seguridad de secuencias pseudoaleatorias binarias procedentes de algoritmos y esquemas criptográficos. El enfoque propuesto para el desarrollo se centra en el estudio e implementación de las diferentes técnicas criptoanalíticas aplicables de acuerdo a la bibliografía, su conjunción y el desarrollo de un conjunto de herramientas que permitan analizar propiedades criptológicas.

2.2 Tecnologías utilizadas

Se trata de una aplicación Web, desarrollada en lenguaje de programación Python versión 3.6, utilizando el *framework* Web Django. Los "análisis", que se integran a la aplicación configurándose como *plugins*, podrían desarrollarse en cualesquier otro lenguaje de programación ya que serán ejecutados como programas externos. Los ejemplos de *plugins* incluidos al momento se encuentran programados en los lenguajes Python y C.

En el repositorio público del proyecto en Github.com se encuentran las instrucciones necesarias para la instalación y puesta en marcha de la herramienta. Dadas las tecnologías mencionadas anteriormente, el software puede ser instalado y ejecutado en diferentes sistemas operativos (Windows, MacOS o Linux).

3 Funcionalidad de la herramienta

En el presente apartado se ejemplificará cómo un usuario podría operar con el sistema. Se describirán a continuación las principales funciones que provee la herramienta actualmente. Nos valdremos de dos muestras de secuencias binarias de aproximadamente la misma longitud; la primera constando del contenido de un archivo comprimido, y la segunda de la salida producida por el archivo de sistema `/dev/urandom` de un sistema Linux, utilizadas sólo como ejemplo y para permitir mostrar mediante capturas de pantalla la operatoria. En el apartado siguiente serán descriptas otras funcionalidades planeadas, a ser implementadas, para completar el alcance propuesto del proyecto.

3.1 Manejo de muestras

Dado que típicamente la primera acción que un usuario realizaría en el sistema sería la agregar muestras a analizar, como puede visualizarse en la figura siguiente, en el menú principal de opciones presentadas a la izquierda de todas las pantallas del sistema, la primera es la de “Muestras”.

Esta opción permite acceder al listado de muestras cargadas. Como fuera mencionado y se observa en la imagen, para ejemplificar el uso de la herramienta, se cargaron dos muestras del tipo “Secuencia binaria”. La primera que si visualiza en el listado corresponde a la salida producida por el archivo de sistema `/dev/urandom` de un sistema Linux, y la segunda al contenido de un archivo comprimido.

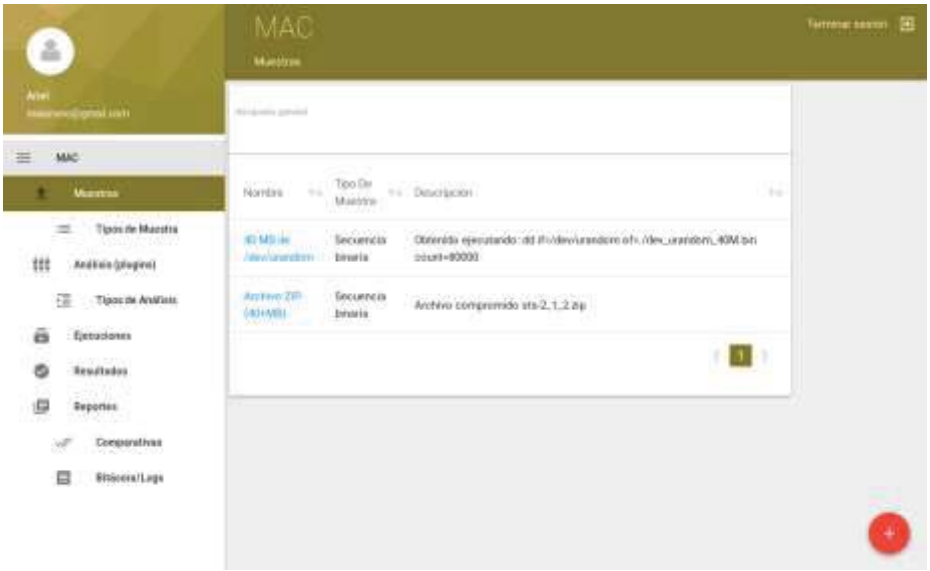

Figura 1. Captura la pantalla del sistema que lista las muestras ingresadas por el usuario para su posterior análisis.

En esta pantalla, utilizando el botón color rojo de agregar a pie de página y alineado a la derecha (símbolo +), se pueden agregar otras muestras a ser utilizadas en los análisis. Esto puede realizarse a partir de un archivo o también introduciendo el valor en formato de texto, pudiendo expresarlo en notación binaria, hexadecimal o codificación ASCII. Se especifica además un tipo de muestra (en ambos casos de ejemplo el tipo corresponde a "Secuencia binaria"). Estos tipos de muestra pueden ser configurados en el sistema utilizando la opción del menú principal "Tipos de muestra". Por último cabe aclarar que el nombre y una descripción opcional se utilizan únicamente para poder referenciar la muestra desde las otras pantallas del sistema que la involucraran.

3.2 Plugins de análisis

La funcionalidad principal del sistema se basa en la aplicación de diferentes técnicas de análisis sobre las muestras disponibles (que hayan sido cargadas en el sistema de acuerdo a lo descripto en el apartado anterior).

Los diferentes análisis que pueden realizarse a través del sistema son configurados como *plugins* o "complementos" (por utilizar una traducción posible).

Estos *plugins* o complementos constan básicamente de, por un lado, programas externos que serán ejecutados por el sistema, y por otro lado, de la configuración necesaria tanto para su ejecución como para la interpretación –y registro en el sistema- de sus resultados.

La siguiente figura muestra el listado de los *plugins* configurados en el sistema:

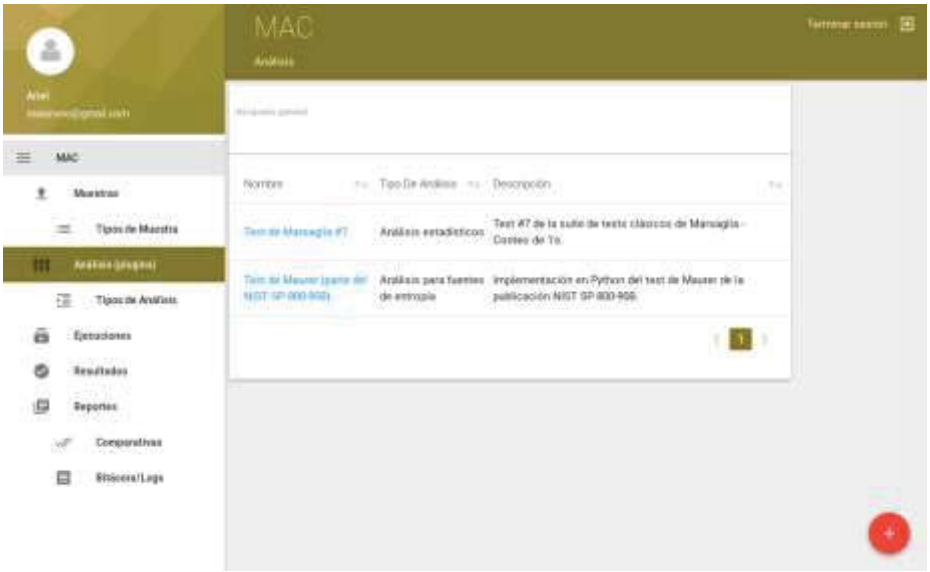

Figura 2. Captura de pantalla del sistema que muestras los *plugins* o "complementos" de análisis configurados.

En este caso, y como se observa en la figura de la captura de pantalla, los *plugins* configurados para la ejemplificación del uso de la herramienta son dos métodos, o testeos de pseudoaleatoriedad, para la realización de análisis estadísticos de secuencias binarias: El séptimo Test de Marsaglia [5] y el Test de Maurer de acuerdo a la publicación SP 800-90B de la NIST [6].

3.2.1. Tests de Marsaglia

Como ejemplo inicial y dentro se incorporó originalmente uno de los llamados Tests de Marsaglia [5], particularmente el identificado con el número siete, también llamado de "conteo de unos".

Este test considera el archivo de muestra como un flujo de bytes. Cada byte puede contener de 0 a 8 dígitos 1 (uno), con probabilidades 1, 8, 28, 56, 70, 56, 28, 8 y 1 sobre 256, respectivamente. Luego considera que el flujo de bytes proporciona un conjunto de palabras de 5 letras, que son supuerpuestas, donde cada letra puede tomar los valores A, B, C, D, E. Las letras se determinan por el número de 1 'en un byte: 0,1 ó 2 resultan en una A, 3 en una B, 4 unos, en una C, 5 en una D y 6,7 u 8 en una letra E. Así el autor originalmente explicó que se tendría un un mono en una máquina de escribir que golpea cinco teclas con varias probabilidades (37, 56, 70, 56 y 37 sobre 256). Hay 5^5 palabras posibles de 5 letras, y de un conjunto de 256,000 palabras de 5 letras (superponiéndose), se hacen conteos de las frecuencias de cada palabra. De lo anterior se deriva una prueba de chi cuadrado. El resultado del testeo que el sistema registra será el valor "p".

3.2.2. Test de Maurer

También considerándolo un ejemplo inicial entre pruebas de esta índole, actualizado en el repositorio de software del proyecto recientemente, la herramienta incorpora una implementación de referencia de la NIST (por sus siglas en inglés de *National Institute of Standards and Technology*) basado en el test de Maurer, o Test Universal de Maurer, o también Estadística Universal de Maurer; según lo especificado en su publicación NIST SP 800-90B, para análisis de fuentes de entropía.

De acuerdo a las especificaciones mencionadas, la prueba de compresión calcula la tasa de entropía de un conjunto de datos –en el caso del sistema, de los archivos de muestra-, según la cantidad de datos que se pueden comprimir. Basado en la Estadística Universal de Maurer, la prueba genera un diccionario de valores y luego calcula el número promedio de muestras requeridas para obtener una salida basada en el diccionario. Una de las ventajas de usar la Estadística de Maurer es que no se asume independencia. Cuando una salida con dependencias se prueba con esta estadística, la tasa de compresión se ve afectada (y, por lo tanto, la entropía), pero aún se obtiene una estimación de la entropía. El cálculo de la Estadística de Maurer requiere solo una "pasada" a través del conjunto de datos para proporcionar una estimación de entropía, por lo que es más eficiente que otros algoritmos de compresión. La estadística de Maurer es la media de los valores de compresión.

3.3 Ejecuciones de análisis

Cargadas las muestras que se desee analizar, y configurados los *plugins* de análisis, el paso siguiente dará un usuario de este sistema será el de ejecución de tales análisis, pruebas o testeos.

Como puede verse en la figura que sigue, mostrando como ejemplo una captura de pantalla habiendo accedido a la opción del menú principal "Ejecuciones", que lista las ejecuciones ingresadas e iniciadas por indicación del usuario en el sistema, se muestran dos ejecuciones finalizadas del análisis del Test de Marsaglia número siete sobre las dos muestras de ejemplo.

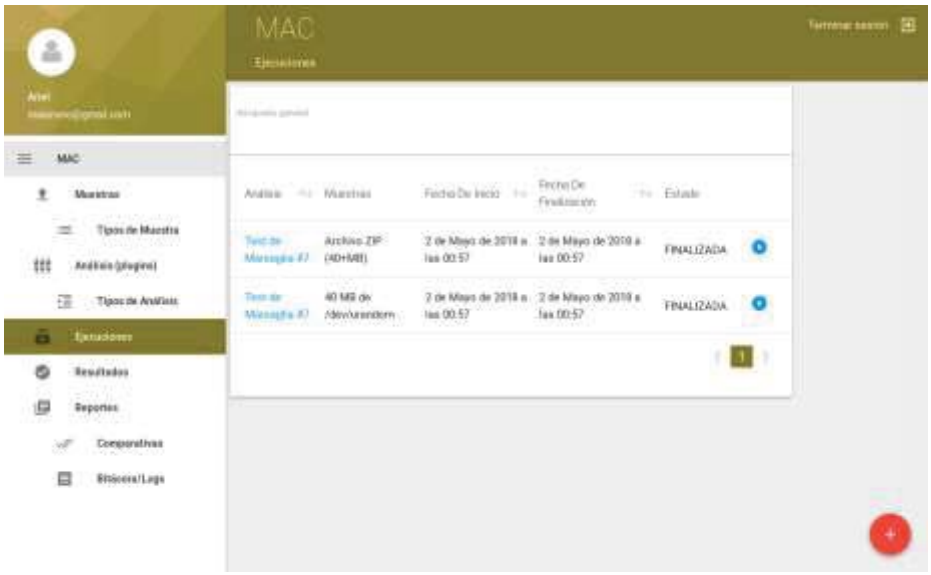

Figura 3. Captura de pantalla donde se visualiza el estado de las ejecuciones de análisis solicitadas por el usuario.

3.4 Presentación de resultados

Finalmente, luego de ejecutados los análisis, el sistema permite por supuesto la visualización de los resultados correspondientes. En primer lugar, a través del listado que puede observarse en la figura a continuación. En segundo lugar, para permitir una vista que facilite la comparación entre los resultados obtenidos, a través de los reportes, que son accesibles a partir de la última entrada de menú en el margen izquierdo de todas las pantallas.

En el ejemplo de la figura puede apreciarse que para el caso de la aplicación del Test de Marsaglia número siete a los dos archivos de muestras de ejemplo disponibles, se visualiza el valor "p" obtenido.

Figura 4. Captura de pantalla en la cual pueden apreciarse los resultados de las ejecuciones de los testeos utilizados como ejemplo.

4 Otras funcionalidades planeadas (Trabajo a futuro)

Se planea la continuación del desarrollo para ampliar la funcionalidad actual de la herramienta, en línea con los objetivos del proyecto.

Entre las funcionalidades a implementar consideradas se encuentran principalmente las relativas a la incorporación de nuevos *plugins* de análisis, para implementar, por ejemplo:

1. Análisis estadísticos adicionales a los ya implementados (secuencias binarias; generadas por generadores pseudoaleatorios –algoritmos de cifrado en flujo, registros desplazables, etc.-) [7,8,9].
2. Análisis de complejidad lineal (secuencias binarias) [7,8].
3. Análisis y ataques por correlación (secuencias binarias) [7,8].
4. Criptoanálisis lineal (algoritmos de cifrado) [9].
5. Criptoanálisis algebraico (algoritmos de cifrado) [10].
6. Criptoanálisis diferencial (algoritmos de cifrado) [11,12].
7. Otras técnicas de criptoanálisis de aparición reciente en la literatura, como ser, por ejemplo, la implementación del "*Cube-Attack*" [13,14] o métodos basados en *SAT-Solvers* [15,16].

5 Conclusiones

Este artículo presentó una herramienta para la aplicación de técnicas criptoanalíticas, configurables mediante *plugins* de análisis. Se describió el contexto del proyecto y sus

objetivos. Llamada MAC, por las siglas de Marco de Aplicación de Criptoanálisis, la herramienta se distribuye como un proyecto de software libre, de código abierto.

Desconocemos la existencia de otras herramientas de acceso libre que implementen la funcionalidad que actualmente implementa y que se planea implementar en nuestro proyecto, específicamente en relación a una interfaz común para el manejo de muestras y la posibilidad de aplicación –y revisión de resultados- de diversos testeos o pruebas de manera modular, en un marco de trabajo común.

También se espera que su publicación pueda lograr que quienes pudieran valerse de una herramienta de este tipo para agilizar y normalizar estos análisis se interesen e involucren en el proyecto.

Referencias

1. Repositorio del proyecto de software MAC en Github. [En línea] https://github.com/gicsi/mac, accedido por últ. vez en julio de 2019.
2. Castro Lechtaler, A., Liporace, J., Cipriano, M., García, E., Maiorano, A., Malvacio, E., Tapia, N. Automated Analysis of Source Code Patches using Machine Learning Algorithms. XXI Congreso Argentino de Ciencias de la Computación (Junín, 2015). ISBN: 978-987-3806-05-6. [En línea] http://sedici.unlp.edu.ar/bitstream/handle/10915/50585/Documento_completo.pdf-PDFA.pdf?sequence=1, accedido por últ. vez en julio de 2019.
3. Proyecto AAP, Repositorio de GICSI en Github. [En línea] https://github.com/gicsi/aap, accedido por últ. vez en julio de 2019.
4. Cipriano, M., Malvacio, E., Estevez, C., Fernández, D., García, E., López, G., Liporace, J., Maiorano, A., Vera Batista, F. Software Abierto para la Evaluación de Sistemas Criptológicos Integrados. XX Workshop de Investigadores en Ciencias de la Computación. RedUNCI, UNNE. ISBN: 978-987-3619-27-4. [En http://sedici.unlp.edu.ar/bitstream/handle/10915/68350/Documento_completo.pdf-PDFA.pdf?sequence=1&isAllowed=y, accedido por últ. vez en julio de 2019.
5. Marsaglia, G. The Marsaglia Random Number CDROM including the Diehard Battery of Tests of Randomness. [En línea] http://www.csis.hku.hk/cisc/download/index2.htm, accedido por últ. vez en julio de 2019.
6. Recommendation for the Entropy Sources Used for Random Bit Generation. Computer Security Resource Center. NIST. [En línea] https://csrc.nist.gov/publications/detail/sp/800-90b/final, accedido por últ. vez en julio de 2019.
7. Massey, J.L. “Shift-register synthesis and BCH decoding”. IEEE Transactions on Information Theory 15, 1969.
8. Golomb. “Shift Register Sequences”. Aegean Park Press, 1982.
9. Christopher Swenson. "Modern Cryptanalysis: Techniques for Advanced Code Breaking". Wiley Publishing. 2012.
10. Nicolas T. Courtois, Karsten Nohl, and Sean O’Neil. Algebraic attacks on the crypto-1 stream cipher in mifare classic and oyster cards. Cryptology ePrint Archive, Report 2008/166, 2008.
11. E. Biham and A. Shamir. "Differential Cryptanalysis of the Full 16-Round DES". In Advances in Cryptology — CRYPTO 1992, volume 740 of Lecture Notes in Computer Science, pages 487–496, 1992.

12. M. Albrecht and C. Cid. Algebraic Techniques in Differential Cryptanalysis. In Fast Software Encryption 2009 – FSE 2009, Lecture Notes in Computer Science. Springer-Verlag, 2009.
13. Dinur, I., Shamir, A.: Cube Attacks on Tweakable Black Box Polynomials. In: EUROCRYPT. pp. 278–299 (2009).
14. Dinur, I., Morawiecki, P., Pieprzyk, J., Srebrny, M., Straus, M.: Cube Attacks and Cube-Attack-Like Cryptanalysis on the Round-Reduced Keccak Sponge Function. In: Advances in Cryptology - EUROCRYPT 2015. 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Sofia, Bulgaria, April 26-30, 2015, Proceedings, Part I. pp. 733–761 (2015).
15. McDonald, C. and Pieprzyk, C. “Attacking Bivium with MiniSat”, Cryptology ePrint Archive, Report 2007/040, 2007.
16. The Z3 Theorem Prover, Repositorio en Github de los proyectos. [En línea] https://github.com/Z3Prover, accedido por últ. vez en mayo de 2017.