Paper Title

Diego Lupi, Pedro Nieto, and Huaira Gómez

FaMAF - Universidad Nacional de Córdoba, Córdoba, Argentina

Abstract. Easycrypt[1] es una herramienta automatizada que soporta la construcción y verificación de pruebas de seguridad de sistemas criptográficos. Permite mejorar la confianza en sistemas criptográficos mediante la entrega de pruebas verificadas formalmente que resultan en sus metas propuestas. Provee una plataforma versátil que soporta pruebas automatizadas pero también permite al usuario realizar puebas complejas de manera interactiva entrelazando la verificación del programa con la formalización de las matemáticas, hecho fundamental al formalizar pruebas criptográficas. Con este paper nos proponemos mostrar las caracteristicas de esta herramienta y compararla con herramientas similares.

Keywords: Easycrypt · Game-based cryptographic proofs · Probabilistic.

1 Introducción

Desde siempre las pruebas criptograficas fueron propensas a errores, lo que naturalmente las puede llevar a ser erróneas. En particular en las pruebas de seguridad criptograficas la correctitud es critica para mejorar la confianza en el sistema criptografico. Actualmente se tiende a generar mas pruebas de seguridad de las que se pueden verificar y se omiten detalles finos desde un analisis formal que pueden tener grandes efectos en la practica. Teniendo en cuenta que los sistemas criptograficos en el mundo real pueden ser vulnerados, es necesario hacer las verificaciones sobre los pruebas de los sistemas criptograficos para evitar un desastre en el area de la seguridad.

Easycrypt es una herramienta automatizada que permite la construccion de pruebas de seguridad de sistemas criptograficos y su verificacion de manera interactiva usando la secuencialidad del codigo con un enfoque de game-based cryptographic proofs. Este enfoque consiste en la interaccion de un retador y un adversario, donde se especifica explicitamente la meta que adversario intenta alcanzar, como por ejemplo suponer de manera correcta una porcion de informacion oculta. En Easycrypt los juegos criptograficos se modelan como modulos, que consisten en procedimientos escritos en lenguaje imperativo. Por otra parte los adversarios se modelan como modulos abstractos, modulos cuyo codigo es desconocido y puede cuantificarse.

Posteriormente se sumo al desarrollo la École Polytechnique (Escuela Politecnica). IMDEA software institute es un instituto para el estudio avanzado de tecnologias para el desarrollo de software asentado en Madrid, España. Inria es

un centro de investigación francés especializado en Ciencias de la Computación, teoría de control y matemáticas aplicadas. Por ultimo, la École Polytechnique es una gran escuela de ingenieros francesa bajo la tutela del Ministerio de Defensa.

El primer prototipo de EasyCrypt lanzado en 2009 fue desarrollado por IMDEA Software Institute, e Inria. Constaba de una interfaz de linea de comando y funcionalidades muy acotadas. Posteriormente se sumo al desarrollo la École Polytechnique (Escuela Politecnica). IMDEA software institute es un instituto para el estudio avanzado de tecnologias para el desarrollo de software asentado en Madrid, España. Inria es un centro de investigación francés especializado en Ciencias de la Computación, teoría de control y matemáticas aplicadas. Por ultimo, la École Polytechnique es una gran escuela de ingenieros francesa bajo la tutela del Ministerio de Defensa. En el año 2012 se le hizo una reimplementacion completa al prototipo con el objetivo de superar varias de las limitaciones que este revelo. Actualmente se encuentra en la version 1.0 que fue liberada el 10 Octubre de 2017. En esta version los desarrolladores permitieron que EasyCrypt pueda ejecutar scripts interactivamente en Proof General[3], dandole a la herramienta una interfaz grafica interactiva en la que el usuario puede simular paso a paso la verificacion de su especificacion, otorgando la posibilidad al usuario de elegir el enfoque por el cual quiere verificar la misma. Por otro lado para proveer las bases requeridas para llevar a cabo algunos razonamientos criptograficos estandares se implementaron cuatro logicas, lo que permite realizar pruebas mas complejas, que en versiones anteriores no eran verificables.

2 Comparación con otras herramientas

Como se nombró anteriormente Easycrypt es una herramienta que se utiliza para la verificación de pruebas de seguridad de sistemas criptográficos, pero Easycrypt no es la única. Para saber sus limitaciones y sus extensiones se comparó con otras herramientas(CertiCrypt, Verypto, FCF). Uno de los principales objetivos es la legibilidad por dos razones. En primer lugar, las definiciones de seguridad deben parecerse a las de la literatura criptográfica, de modo que los criptógrafos puedan comprenderlas y evaluarlas. Segundo, la sintaxis intuitiva apoya a los usuarios durante la formalización, ya que pueden enfocarse en formalizar los argumentos en lugar de que tratar de entender programas poco legibles, todos los marcos logran buena legibilidad, pero de diferentes maneras. CertiCrypt y EasyCrypt se utiliza un lenguaje de procedimiento imperativo en sus lógicas. Ellos modelan de cerca la idea de Bellare y Rogaway de un lenguaje con estado con oráculos como procedimientos[4]. Verypto integra profundamente un lenguaje de orden superior con referencias mutables basadas en índices de Bruijn en HOL[5]. La legibilidad se recupera al reflejar la sintaxis en el lenguaje de términos de HOL mediante el análisis y pretty-printing tricks. Por el contrario, FCF y nuestro marco insertan superficialmente un lenguaje funcional con secuenciación monádica en Coq e Isabelle/HOL, respectivamente[6]. Otro objetivo es la expresividad, la misma tiene dos dimensiones. Primero, la sintaxis y el dominio semántico determinan la expresividad del lenguaje, CertiCrypt y EasyCrypt admiten distribuciones disc-

retas de subprobabilidad y un lenguaje de procedimiento, pero no un operador de punto fijo. Los puntos fijos no aumentan la expresividad en un combinador de tiempo, sino que conducen a formulaciones de programas más naturales. EasyCrypt también proporciona un sistema modular para admitir la abstracción y la reutilización. Verypto es el marco más general, ya que se basa en la teoría de medidas y, por lo tanto, admite distribuciones continuas y funciones de orden superior. La semántica de FCF permite solo distribuciones de probabilidad con soporte finito, por lo que no se puede definir un operador de punto fijo. La sintaxis restringe aún más los efectos probabilísticos al muestreo aleatorio de cadenas de bits y probabilidades condicionales. En segundo lugar, la lógica determinan qué tipo de propiedades de seguridad se pueden formalizar. Todos los marcos admiten pruebas de seguridad concretas. Gracias a la integración profunda, CertiCrypt y Verypto también admiten declaraciones sobre la eficacia y, por lo tanto, declaraciones de seguridad asintóticas, lo cual es imposible tanto para EasyCrypt, ya que las funciones de HOL son extensiones. FCF viene con un modelo de costo axiomático para la eficiencia, que no está formalmente conectado a un modelo operativo. Otro objetivo fundamental es la confiabilidad, el código confiable influye en la confiabilidad de una prueba mecanizada y debe ser lo más pequeña posible. Los asistentes de prueba como Coq e Isabelle consisten en un kernel y además pueden producir términos de prueba y objetos de prueba que un verificador independiente puede certificar. EasyCrypt no tiene un kernel, por lo que debe confiarse en la implementación completa en OCaml y en los SMT solvers lo cual es una desventaja.

EasyCrypt es el sucesor de CertiCrypt ya que presenta un mecanismo de compilar pruebas EasyCrypt en pruebas de CertiCrypt y ademas en EasyCrypt el esfuerzo humano es menor como se ve la fig.1. En fig.1 se compara CertiCrypt con EasyCrypt en varias pruebas de seguridad formalizadas en ambos sistemas. Los tiempos se miden en una Intel Core 2 Duo con 2.8GHz y 4GB de RAM sobre Mac OS X 10.6.7.

	CertiCrypt		*EasyCrypt*		*Extracted*	
	Lines	Time	Lines	Time	Lines	Time
ElGamal $(IND - CPA)$	565	45s	190	12s	1130	23s
Hashed ElGamal $(IND - CPA)$	1255	1 m 5s	243	33s	1772	41s
Full-Domain Hash $(EF - CMA)$	2035	5 m 46s	509	1 m 26s	2724	1 m 11s
Cramer-Shoup $(IND - CCA)$	n/a	n/a	1637	5 m 12s	5504	3 m 14s
OAEP $(IND - CPA)$	2451	3 m 27s	n/a	n/a	n/a	n/a
OAEP $(IND - CCA)$	11162	37 m 32s	n/a	n/a	n/a	n/a

Fig. 1.

References

1. Gilles Barthe, Juan Manuel Crespo, Benjamin Gregoire, Cesar Kunz, Santiago Zanella Beguelin. Computer-Aided Cryptographic Proofs. Third International Con-

ference, 2012.
2. OCaml Website: (2013) https://ocaml.org.
3. Proof-General Website: (2016) https://proofgeneral.github.io.
4. Bellare, M., Rogaway, P.: The security of triple encryption and a framework for code-based game-playing proofs. In: EUROCRYPT 2006. LNCS, vol. 4004, pp. 409–426. Springer (2006)
5. Backes, M., Berg, M., Unruh, D.: A formal language for cryptographic pseudocode. In: LPAR 2008. LNCS, vol. 5330, pp. 353–376. Springer (2008)
6. Shoup, V.: Sequences of games: A tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332 (2004)