Cryptol: A Domain Specific Language for Verification of Cryptographic Algorithms

Zhiyuan Lin

July 19, 2016

Abstract

Contents

Chapter 1

Introduction

1.1 Organizations

Chapter 2

Literature Survey

In this chapter we present previous works that uses programming language as a means to improve reliability of cryptographic applications. In Section 2.1, we give broad overview of several different works, whereas in Section 2.2 we dig into the details of the Cryptol programming language and the guarantees it provides.

2.1 Languages for Cryptographic Applications

The idea of using language features to enhance cryptogprahic applications have been investigated for over a decade. Some works, such as [1] builds upon existing languages, and provides extensions, e.g. libraries and frameworks, for efficient implementation of cryptographic protocols. Other studies [2] create brand-new domain specific programming languages dedicated to cryptographic applications. These works also focus on different aspects of implementation. Some focus on relaibility and correctness guarantees, while others emphasize ease of use and performance.

Charm [1] is an extensible framework in Python designed for rapid prototyping of cryptographic schemes. Charm promotes modularity and reusability of cryptographic primitives, and successfully increases inter-operability of existing numeric libraries such Sage and the Stanford Pairing-Based Crypto (PBC). It also provides benchmarking and profiling utilities for determining the performance of cryptographic algorithms.

NaCl [3] is a C/C++ library for implementing cryptographic protocols that provides security guarantee through features such as no data flow from secrets to load address, and no padding oracles.

ZKPDL [4] is an interpreted description language for specifying zero-knowledge protocols, motivated by applications such as electronic cash. Although the language is designed specifically for implementing prover and verifer of zero-knowledge, the language itself also have potentials for specifying other types of privacy-preserving systems. The ZKPDL interpreter also performs optimizations for protocols.

EasyCrypt [5] is an interactive framework for verifying the security of cryptographic applications. Building on EasyCrypt, The Ceritified computer-aided cryptography [6] project provides a computer-aided framework for proving concrete security for cryptogprahic implementations. It extends the EasyCrypt framework to provide formal verification for cryptographic applications implemented in a C-like language. The framework also supports generation of optimized machine code based on the high-level language while retaining the security properties.

2.2 The Cryptol Language

Chapter 3

Implementation and Empirical Evaluation

This section presents the implementation part of the project.

3.1 Design

3.2 High Assurance Programming with Cryptol

The aim of this project is to investigate domain specific languages that enhance reliability of cryptographic algorithms. The focus of this work is on Cryptol, however other languages with similar facilities will also be covered as related works.

May 25th - June 2nd:	Surveying Cyptol and similar languages
June 2nd - June 9th:	In-depth investigation of Cryptol's design and theory
June 9th - June 23rd:	Implementing AES algorithm in Cryptol
June 23rd - July 30th:	Verification of implementaion using Cryptol and SAW.
July 30th - July 7th:	Finishing up evaluation and writing report
July 7th - July 14th:	Preparing for presentation

Figure 3.1: Time Line of The Project

Therefore the first part of the project will be a brief survey of Cryptol and similar languages. This part will also cover high-level design of the Cryptol language and theory behind its functions for formal verification.

The second part of the project will be empirical study of the Cryptol language with an implementation. For the implementation part, the AES symmetric-key algorithm [?, ?] is to be implemented and properties related to the algorithm will be defined and checked in Cryptol to evaluate whether the language can efficiently verify the implementation. AES is chosen because it is the modern standard for symmetric-key encryption is widely used. The focus in this part is the evaluation of the language, rather than actual implementation of the algorithm, therefore other algorithms written in Cryptol, if available, will also be used to conduct empirical evaluations.

Another tool provided as a part of the Cryptol project is called The Software Analysis Workbench (SAW). SAW also provides formal verification for properties of programs written in Cryptol. SAW utilizes symbolic execution to translate programs into formal models. This tool will also be used to verify the AES implementation in Cryptol in order to see if it provides better functionalities for verification.

Figure 3.1 provides a time frame for the project. This is just a rough estimation, but the project will follow the steps specified. As mentioned before, because the focus is investigation of the Cryptol language, more time will be spent on evaluating the language functionalities.

Chapter 4

Conclusion and Future Works

Bibliography

[1] J. A. Akinyele, C. Garman, I. Miers, M. W. Pagano, M. Rushanan, M. Green, and A. D. Rubin, "Charm: a framework for rapidly prototyping cryptosystems," *Journal of Cryptographic Engineering*, vol. 3, no. 2, pp. 111–128, 2013.

[2] J. R. Lewis and B. Martin, "Cryptol: High assurance, retargetable crypto development and validation," in *Military Communications Conference, 2003. MILCOM'03. 2003 IEEE*, vol. 2, pp. 820–825, IEEE, 2003.

[3] D. J. Bernstein, T. Lange, and P. Schwabe, "The security impact of a new cryptographic library," in *International Conference on Cryptology and Information Security in Latin America*, pp. 159–176, Springer, 2012.

[4] S. Meiklejohn, C. C. Erway, A. Küpçü, T. Hinkle, and A. Lysyanskaya, "Zkpdl: A language-based system for efficient zero-knowledge proofs and electronic cash.," in *USENIX Security Symposium*, vol. 10, pp. 193–206, 2010.

[5] G. Barthe, F. Dupressoir, B. Grégoire, C. Kunz, B. Schmidt, and P.-Y. Strub, "Easycrypt: A tutorial," in *Foundations of Security Analysis and Design VII*, pp. 146–166, Springer, 2014.

[6] J. B. Almeida, M. Barbosa, G. Barthe, and F. Dupressoir, "Certified computer-aided cryptography: efficient provably secure machine code from high-level implementations," in *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pp. 1217–1230, ACM, 2013.