Verifying Cryptographic Implementations with Cryptol

•••

Zane Zhiyuan Lin

Implementing Cryptographic Algorithms Is Hard

- Specification is not always clear or complete.
- Involves deep mathematical theory.
- Implementation details could compromise security.
    - It is assessed that millions of HTTPS, SSH, and VPN servers use the same prime numbers for Diffie-Hellman key exchange.
- “Don’t roll your own crypto.”

The Cryptol Language

Cryptol is a domain-specific language for implementing cryptographic algorithms.

- Cryptol is written in Haskell, and just like Haskell it is a pure functional programming language.
- Cryptographic algorithms implemented in Cryptol is close to their mathematical specification and can be used as reference implementation.
- Cryptol offers high-assurance of correctness.

High-Assurance Programming in Cryptol

Type System

- Types in Cryptol express the size and shape of data

Formal Verification

- Cryptol allows for specifying and formally proving correctness properties

Automated Testing

- In cases where formal verification is too slow, properties can also be tested automatically

Example

Below is an implementation of Caesar cipher in Cryptol:

```
caesar : {n} ([8], String n) -> String n
caesar (s, msg) = [ shift x | x <- msg ]
        where
                map = ['A' .. 'Z'] <<< s
                shift c = map @ (c - 'A')

decryptCaesar: {n} ([8], String n) -> String n
```

The correctness property for this implementation expressed in Cryptol:

```
property caesarCorrect (d, msg) =
        if validMessage msg
        then decryptCaesar (d, caesar(d, msg)) == msg
        else True
```

Implementing and Verifying AES

Strategy:

- Instead of verifying the entire implementation at one go, verify the correctness of each component.
- Use automated testing where verification is too slow.

Results:

Function	Correctness Proof
AddRoundKey	passed
SubBytes	passed
ShiftRows	passed
MixColumns	passed
decrypt(encrypt(m, k), k) == m	Does not terminate. Automated testing passed.

Conclusion

- Cryptol provides useful tools for specifying and verifying cryptographic algorithms.
- Implementations in Cryptol can also be used to verify crypto algorithms in other languages (check out *Software Analysis Workbench*)
- Drawbacks:
  - Lack of support for floating point numbers
  - Missing language features and library support (compared to Haskell)
  - Correctness is not security

References

- Lewis, J.R. and Martin, B., 2003, October. Cryptol: High assurance, retargetable crypto development and validation. In *Military Communications Conference, 2003. MILCOM'03. 2003 IEEE* (Vol. 2, pp. 820-825). IEEE.
- Erkök, L. and Matthews, J., 2009, April. High assurance programming in Cryptol. In *Proceedings of the 5th Annual Workshop on Cyber Security and Information Intelligence Research: Cyber Security and Information Intelligence Challenges and Strategies* (p. 60). ACM.
- Galois, Inc., 2010, *Cryptol: The Language of Cryptography*. Galois, Inc.
- Daemen, J. and Rijmen, V., 2013. *The design of Rijndael: AES-the advanced encryption standard*. Springer Science & Business Media.

Thank You!