第 38 卷 第 3 期
Vol.38 No.3

计 算 机 工 程
Computer Engineering

2012 年 2 月
February 2012

• 安全技术 •

文章编号：1000—3428(2012)03—0121—03 文献标识码：A 中图分类号：TP319

基于 Hoare 逻辑的密码软件形式化验证系统

郝耀辉，郭渊博，罗 婷，燕菊维

(解放军信息工程大学电子技术学院，郑州 450004)

摘 要：在 Hoare 逻辑理论和 ACSL 语法规范的基础上，设计一种针对密码软件的形式化验证系统，由程序规范、验证推理规则、可靠性策略、验证推理等模块组成。以 OpenSSL 中 RC4 算法的软件实现为例，对其功能正确性、保险性和信息流安全性进行验证，结果表明，该系统具有较高的自动化水平，可在一定程度上降低形式化验证方法的复杂度。

关键词：Hoare 逻辑；密码软件；形式化验证；程序规范；RC4 算法

Formal Verification System of Cryptographic Software Based on Hoare Logic

HAO Yao-hui, GUO Yuan-bo, LUO Ting, YAN Ju-wei

(Institute of Electronic Technology, PLA Information Engineering University, Zhengzhou 450004, China)

【Abstract】Based on Hoare logic and ANSI/ISO C Specification Language(ACSL) specification, this paper presents a formal verification system for cryptographic software, which is composed of program specification, inference rules, reliability strategy and verification module. It takes the software realization of RC4 algorithm in OpenSSL as an example, the functional correctness, safety properties and information flow security are tested and verified. Results show that this system can reduce the complexity of formal verification method and has a high level of automation.

【Key words】Hoare logic; cryptographic software; formal verification; program specification; RC4 algorithm

DOI: 10.3969/j.issn.1000-3428.2012.03.041

1 概述

密码模块是保障安全系统中信息机密性与完整性的重要部分，在许多安全系统中，密码模块主要是由密码算法的软件实现构成，即密码软件部分。这就要求密码软件在向系统提供安全服务的同时，其自身的安全性也应得到保证。

对此美国国家标准技术局(NIST)和加拿大通信安全局(CSE)提出 CMVP(Cryptographic Module Validation Program)计划，规定了对安全软件的验证准则 DTR[1](Derived Test Requirements)。但 CMVP 计划主要依赖验证者的经验，完全依靠手工完成，存在出错率较高、验证周期长、效率低等缺点，其时效性和完备性已满足不了实际应用的需求。为此，本文基于 Hoare 逻辑理论，提出一种密码软件的形式化验证系统。

2 相关知识

本文主要基于 Hoare 逻辑原理，依据 ACSL(ANSI/ISOC Specification Language)语法规范，对待验证的密码软件添加满足其需要验证的关键特性的前置、后置条件，再用所设计的验证系统对其进行验证。

2.1 Hoare 逻辑

Hoare 逻辑[2]是广泛应用的对命令式语言程序进行推理验证的逻辑系统，其基本思想是在代码段与调用者之间构建一种合同似的规格说明(contracts)，用于描述一段代码执行前后计算机状态的改变情况，由一个前置条件和一个后置条件构成，表示形式为：{Pre}P{Post}，称为 Hoare 三元组或断言。其中，Pre 是前置条件，又称初始断言，描述代码段执行前程序状态必须满足的条件，即输入值必须具有的性质；Post 是后置条件，又称终结断言，描述在代码段正确运行后程序状态所需要满足的条件，即输出值应该具有的性质。

2.2 ACSL 语言

ACSL[3]是一种以注释形式加在程序代码中，专门用于描述程序性质的形式化语言。该语言主要以函数合约(function contract)的形式存在，即要求对任一函数 f，需明确描述清楚函数 f 开始时(输入)参数值的要求和结束时(输出)返回值应具有的性质。

其涵义是：若调用函数 f 前，前置条件成立，则函数 f 执行完后，后置条件也必须成立。其实质和 Hoare 三元组表示的内容等同。

2.3 密码软件的关键特性

通过分析密码软件的特性，对保障密码软件安全的至关重要属性进行归纳总结，主要将其归为功能正确性、保险性、信息流安全性 3 类属性[4]，下面对其进行说明。

2.3.1 功能正确性

主要保证密码软件中程序的执行符合相应的设计规范，简单的说就是保证程序执行的输入、输出行为和设计规范相匹配。本系统将其用于验证密码软件输出值是否符合项目输入值和输出值之间的关系。

2.3.2 保险性

主要指密码软件运行时不引起危险、灾难的能力，本文系统中主要将其用于验证密码软件在开发过程中是否存在缓

基金项目：国家“863”计划基金资助项目“基于规范的容忍入侵中间件关键技术与平台”(2007AA01Z405)；河南省科技创新杰出青年计划基金资助项目(104100510025)

作者简介：郝耀辉(1978－)，女，讲师、硕士，主研方向：信息安全，密码学，数据库技术；郭渊博，副教授、博士；罗 婷，硕士研究生；燕菊维，助教、硕士

收稿日期：2011-05-17 **E-mail：**hao_yaohui@126.com

冲区溢出、数组访问越界、悬空指针访问、变量未初始化等错误。

2.3.3 信息流安全性

主要指密码软件能否保护重要数据的机密性和完整性，使窃密者无法根据其对程序运行行为的观察获取或推断程序中的重要(更高级)数据。即保证高密级输入变量和低密级输出变量之间是非干扰的，从而保证密码软件的运行不会引起泄密行为发生。

3 系统设计与实现

3.1 系统总体结构

在模块化构造软件系统的指导下，以 Hoare 逻辑和 ACSL 语法规范为基础，进行形式化验证系统[5]的设计，该系统主要由程序规范、验证推理规则、可靠性策略[6]、验证推理模块等组成，其总体结构如图 1 所示。

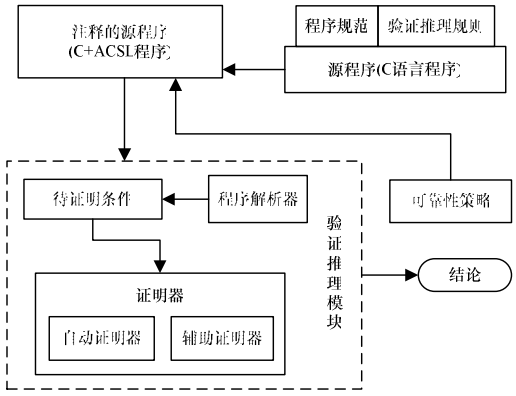

图 1 系统总体结构

(1)程序规范：指密码软件编写时的依据，在实际应用中，通常以操作描述的方式给出，程序员在实现源代码时，必须保证密码软件的输入、输出等与规范(密码算法)规定的一致。

(2)验证推理规则：是系统对密码软件进行验证推理的理论依据，主要包括 Hoare 逻辑理论、ACSL 语法规范等。

(3)可靠性策略：是密码软件应遵循的原则，是对程序添加注释时的依据，也是验证推理模块进行判断的依据。主要包括密钥长度的大小、内存的范围等。

(4)注释的源程序：针对需要验证的密码软件的特有特性，如功能正确性、保险性、信息流安全性等，在 Hoare 逻辑基础上，依据 ACSL 语法规则添加的注释语句是验证推理模块进行验证的基础。

(5)验证推理模块：对密码软件特有特性进行验证，如果添加的注释程序中的验证条件经验证证明是正确的，则能够说明程序是符合程序规范和可靠性策略的，否则不符合。

(6)程序解析器：收集程序中的规范注释信息，并将其解析，生成待证明的验证条件，由证明器对此进行证明。

(7)待证明条件(Verification Condition, VC)：是指在证明程序满足规范的过程中，根据程序及其规范经过推理产生的一些逻辑断言，如果通过证明器能证明这些逻辑断言为真，则说明程序是满足规范要求的。

(8)证明器：本系统的证明器由自动证明器和辅助证明器组成，先由自动证明器对待证明条件进行证明，对需要人参与交互的，需由使用人员使用辅助证明器对证明过程进行人为参与。

3.2 系统工作流程

使用本文系统对密码软件功能正确性、保险性或信息流安全性进行验证的过程如图 2 所示。

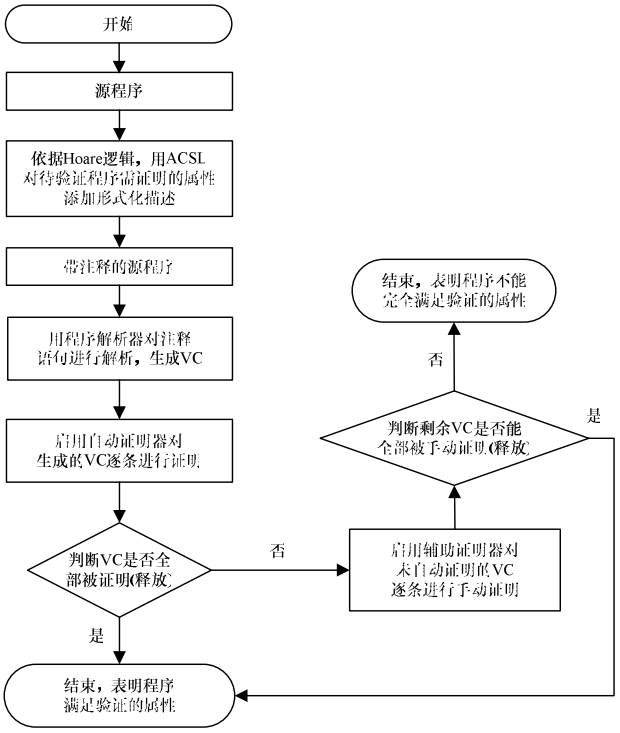

图 2 系统工作流程

工作过程描述如下：

(1)依据 Hoare 逻辑，采用 ACSL 对所要验证的密码软件的待证明属性进行形式化描述，即对源程序添加注释，注释主要包括公理、引理、推理规则等，生成带注释的源程序。

(2)调用程序解析器对程序中的注释语句进行解析，生成待证明条件。

(3)启用自动证明器对生成的 VC 逐条进行证明。

(4)判断 VC 是否全部被证明(释放)，如果 VC 全部被释放，表明程序满足验证的属性，证明结束；否则，转步骤(5)。

(5)启用辅助证明器对未自动证明的 VC 逐条进行手动证明。

(6)判断用辅助证明器证明的 VC 是否全部被释放，如果 VC 全部被释放，则表明程序满足验证的属性，证明结束；否则，表明程序不能完全满足待验证的属性。

3.3 验证推理模块的实现

基于上文提出的形式化验证系统，依据其工作流程，在开源工具 Frama-c[7]、Alt-ergo、Simplify、Z3、Coq[8]等基础上，对其中的核心部件验证推理模块进行设计和实现，其内部详细结构如图 3 所示。

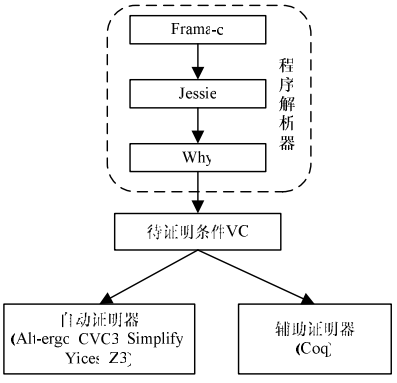

图 3 验证推理模块的内部结构

由图 3 可看出，验证推理模块主要通过程序解析器对程序中的形式化注释语句进行解析，并生成待证明条件。

程序解析器主要是在 Frama-c 这个开源平台协作框架的基础上，加装形式化验证插件 Jessie 实现。首先 Frama-c 内核编译添加注释后的程序；然后调用 Jessie 插件，Jessie 插件基于 Hoare 逻辑中的 Dijkstra 最弱前置条件演算对经 Frama-c 内核编译通过的代码和注释进行演绎推理，判断代码和注释是否符合 Hoare 逻辑的规格说明；最后调用一种基于 Hoare 逻辑的形式化解释程序正确的工具 Why，编译上述注释语句，将其转化为一系列的待证明条件。

待证明条件最后由自动证明器(Alt-ergo, CVC3, Simplify, Yices, Z3)对其进行自动消解或由辅助证明器(Coq)在人工参与的情况下对其进行消解，如果能全部被释放，则表明程序满足验证的属性，否则表明程序不能完全满足待验证的属性。

4 实验结果与分析

本文选用 OpenSSL 中 RC4 算法的具体实现为实例，使用本文设计的验证系统，对其保险性、信息流安全性、功能正确性 3 个特性进行形式化验证。实验的硬件环境为 AMD Phenom(tm) 8650 Triple-Core Processor 2.31 GHz 的 CPU 和 2.00 GB 的内存；软件环境使用的是 Ubuntu Linux 操作系统，内核为 2.6.32。实验产生的具体数据如表 1 所示。

表 1 验证 OpenSSL 中 RC4 算法的具体数据

指标	添加引理/公理数目	产生待证明条件的数目	自动消解待证明条件的数目
保险性	13	881	874
信息流安全性	7	73	44
功能正确性	9	96	70

由此可以看出，本文验证系统可对产生的待证明定理实现大部分的自动证明，其自动证明率达到 93.6%，充分说明该验证系统具有一定自动化形式验证性能，可快速高效地检测密码软件是否满足其安全需求。

5 结束语

本文设计并提出了一种密码软件的形式化验证系统，可对密码软件的功能正确性、保险性和信息流安全性等特性进行自动的形式化验证，并通过对 OpenSSL 中 RC4 算法软件实现的实例测试，证实该系统可完成大部分的自动化证明。但其中使用的验证推理规则等内容仍需人为参与。因此，本文验证系统还不是一套完整的自动化形式化验证系统，还有许多内容需要完善，这将是后续工作的重点。

参考文献

[1] Havener W, Medlock R, Mitchell R, et al. Derived Test Requirements for FIPS PUB 140-2[R]. [S. l.]: National Institute of Standards and Technology, Tech. Rep.: FIPS 140-2, 2004.

[2] 杨 静. 用 Hoare 逻辑验证程序的一般方法及实例[J]. 通讯和计算机, 2007, 4(2): 79-81.

[3] Baudin P, Cuoq P, Jean-Christophe F, et al. ACSL: ANSI/ISO C Specification Language Version 1.5[EB/OL]. [2011-02-21]. http://frama-c.com/download/acsl.pdf.

[4] Almeida J B, Barbosa M, Pinto J S, et al. Deductive Verification of Cryptographic Software[J]. Innovations in Systems and Software Engineering, 2010, 6(3): 203-218.

[5] 李兆鹏，陈意云，葛 琳，等. 一种汇编程序的形式验证框架[J]. 计算机研究与发展, 2008, 45(5): 825-833.

[6] Vieira B. Formal Verification of Security Policies of Cryptographic Software[EB/OL]. (2010-09-03). http://www3.dsi.uminho.pt/seeum2010/CD/abstracts/2165-4.pdf.

[7] Correnson L, Cuoq P, Puccetti A. Frama-c User Manual[EB/OL]. (2011-02-01). http://frama-c.com/download/frama-c-user-manual.pdf.

[8] The Coq Development Team. The Coq Proof Assistant Reference Manual[EB/OL]. (2010-12-23). http://coq.inria.fr/distrib/V8.3pl1/files/Reference-Manual.pdf.

编辑 陆燕菲

(上接第 120 页)

型中的 *GI*、*TI* 和本地域相关的凭证和属性信息；IdP_{Temp} 生成 *TI*，并存储用户访问服务所需的凭证和属性信息。

4 结束语

如何构建普适于现有身份管理架构或应用方式的通用身份模型成为制约身份管理技术发展和应用的一个瓶颈。本文在考虑身份管理需求的基础上，提出一个通用身份模型，给出构建身份模型的方法。该模型对身份管理中的用户隐私保护及信任管理 2 个方面的问题考虑不够，在后续工作中将着重完善模型对这 2 个功能的支持。另外，构建基于通用身份模型的管理架构以及在身份管理架构中实现细粒度的授权管理也是后续工作的重点。

参考文献

[1] Bertino E, Paci F, Shang N. Digital Identity Protection-concepts and Issues[C]//Proc. of ARES'09. Fukuoka, Japan: [s. n.], 2009.

[2] Josang A, AlZomai M, Suriadi S. Usability and Privacy in Identity Management Architectures[C]//Proc. of Australasian Information Security Workshop. Ballarat, Australia: [s. n.], 2007.

[3] Maliki E I, Seigneur T. A Survey of User-centric Identity Management Technologies[C]//Proc. of IARIA'07. Valencia, Spain: [s. n.], 2007.

[4] Damiani E, Vimercati D C D, Samarati P. Managing Multiple and Dependable Identities[J]. IEEE Internet Computing, 2003, 7(6): 29-37.

[5] McLaughlin M P, Jennings B. A Model for Identity in Digital Ecosystems[C]//Proc. of DEST'09. Istanbul, Turkey: IEEE Press, 2009: 295-300.

[6] Cameron K. The Law of Identity[EB/OL]. (2005-04-11). http://ts-si.org/files/TheLawsOfIdentity.pdf.

[7] Cao Yuan, Yang Lin. A Survey of Identity Management Technology[C]//Proc. of ICITIS'10. Beijing, China: [s. n.], 2010.

[8] 张红旗，张文波，张 斌，等. 网格环境下基于身份的跨域认证研究[J]. 计算机工程, 2009, 35(17): 160-162.

编辑 陆燕菲