CertiCore实验设计

谢兴宇2017011326　　韩志磊2017011442

2020年3月29日

1 操作系统的形式化验证

使用形式化验证的技术，对OS进行验证已有很长的一段历史。目前主要有三种验证的技术：

1. 交互式定理证明。这种方法使用定理证明器(Proof Assistant)如Coq辅助验证，本质上是手动证明，利用证明器的特性可以实现一部分自动化。这种方法要求在形式系统内进行演绎，难度很高。典型的系统包括seL4和CertiKOS。

2. 使用程序标注辅助证明。通过在程序中显式地插入约束、不变量等，利用证明器自动将其转化为约束求解问题进行求解。这种方式的自动化程度更高，但是手动标注依旧很困难。典型的系统如Komodo。

3. 完全自动化的求解。这种方式期望验证者集中在编写实现接口、规范上，证明器将通过符号执行过程，自动生成约束，问题最终将被转化为可满足性的判定问题。这种方式简单，但是对实现有一些限制，如要求方法必须是有限的。典型的系统是HyperKernel。

Xi Wang[1]等人按照符号执行的思想，开发了Serval框架。此框架实现了状态机精化，符号执行优化等过程，大大简化了系统验证的开销。我们希望在此框架的基础上，完成对Ucore的验证。

1.1 Serval的实现

Serval基于Rosette语言实现，Rosette本身已经提供了一部分符号执行特性。虽然一阶谓词逻辑的可满足性是不可判定的，但是Rosette实现的是其中可判定的子集。因此，编写程序规范时，可能会需要一部分技巧来规避表达力不足的问题。

Serval在此基础上提供了更高级的特性。使用Serval验证系统的基本思路是：

1. 使用Rosette编写一个automated verifier，将系统的实现代码（如汇编，LLVM IR等）转换为符号执行

2. 使用Rosette编写程序规范

Serval会使用Rosette生成SMT约束，并在此过程中自动进行优化。验证者无需关心验证的过程，只需专注在程序接口和规范的编写。

1.2 规范

Serval提供了准确的状态机精化支持。在Serval中，规范由四部份组成，都在Rosette中编写：

1. 程序的抽象状态s
2. 程序的预期运行行为f_{spec}
3. 一个将程序具体状态映射到抽象状态的abstract function（AF）
4. 程序执行前后的不变量RI

假设automated verifier运行的转移函数是f_{impl}，那么Serval的状态机精化目标是检查下式成立：

$$\forall c((RI(c) \wedge AF(c) = s) \rightarrow (RI(f_{impl}(c)) \wedge AF(f_{impl}(c)) = f_{spec}(s)))$$

验证通过后，我们就可以用抽象状态编写更多的规范了。比如，欲验证状态的转移与某变量无关(noninterference)，为此，我们只需考虑抽象状态的转移。Serval提供了大量的内置方法刻画常见的程序属性，在这个例子中，可以使用其提供的step consistency属性。我们只需要在状态空间上定义一个二元谓词~描述状态间的等价关系，调用Serval的接口后，其会验证下式成立：

$$s_1 \sim s_2 \Rightarrow f_{spec}(s_1) \sim f_{spec}(s_2)$$

因此，用Serval进行验证是较为方便的。

2 CertiCore

我们计划验证UCore。考虑到操作系统的复杂性，我们和riscv64-ucore组进行了沟通，希望按照UCore原本的实验步骤，以及他们编写的step-by-step文档进行验证。

目前，我们已经调研了Serval相关的工作，复现了Serval对CertiKOS、Komodo等系统的验证结果。我们有如下的计划：

- 学习Rosette、x86-32和LLVM IR的验证器
- 模仿Serval对CertiKOS等系统的验证过程，验证一个简单的玩具系统
- 按照实验手册进行UCore的验证

我们希望能为UCore验证OS的常见属性，如进程隔离、内存有限等等。

参考文献

[1] Luke Nelson, James Bornholt, Ronghui Gu, Andrew Baumann, Emina Torlak, 与 Xi Wang. Scaling symbolic evaluation for automated verification of systems code with Serval. In Proceedings of the 27th ACM Symposium on Operating Systems Principles - SOSP '19, pages 225–242. Huntsville, Ontario, Canada, 2019. ACM Press.