УДК 004.056.55

Е.Г. КАЧКО, канд. техн. наук, Д.К. ТЕЛЕВНЫЙ

ИССЛЕДОВАНИЕ ПРИМЕНИМОСТИ SMT/SAT ДОКАЗАТЕЛЬСТВ В КРИПТОАНАЛИЗЕ ХЕШ-ФУНКЦИЙ СЕМЕЙСТВА КЕССАК

Введение

Большой областью применения средств доказательства SMT (*SMT solvers*) является генерация тестовых случаев, статический анализ программ, ограниченная проверка моделей (*BMC*) и *k*-индукция, проверка на эквивалентность.

SMT (*Satisfiability modulo theory)* это обобщение Boolean SAT (*Satisfiability theory*) [1], где набор переменных замещается предикатами из теорий. Одними из таких теорий являются теории структур данных, применяемых при верификации программ. Особую роль в криптологии имеют теории массивов и битовых векторов, применяющиеся при моделировании функций.

В настоящее время существует множество SMT решателей: CVC, CVC3, Z3, многие выпущены под свободными лицензиями (BSD, MIT).

Основная идея исследования

Проблема выполнимости – задача определения – имеет ли формула модель. В булевых выражениях такой моделью является присваивание истинности булевым переменным. В логике предикатов первого порядка модель присваивает значения из домена переменным и доменных интерпретаций функциям и предикатным символам. Для теорий, таких как арифметические или теории структур данных, модель принимает определенное множество интерпретаций символов теории.

В компьютерных науках и прикладной математике SMT проблема – проблема нахождения решения для логических формул на основе теорий, описанных логикой первого порядка.

В криптологии применение SMT сводится к валидации путем доказательств выполнимости или невыполнимости формул F в теориях. Теории первого порядка являют собой множество дедуктивно-полных высказываний.

Применительно к криптоанализу SMT можно использовать для нахождения выполнимых или невыполнимых формул на основе теорий, для групп криптографических преобразований. Таким образом, при моделировании криптопреобразований с применением SMT возможно не только тестирование, но и нахождение возможных слабых мест.

Криптографические модели строятся на алгебраических и логических преобразованиях.

Рассмотрим основные определения для доказательств, основанных на теориях. Пусть *DC(G)* дедуктивное замыкание множества высказываний *G*. Тогда для каждой теории $DC(\tau) = \tau$. Теория называется совместимой если $\text{False} \notin \tau$.

Формула $\phi(x)$ выполнима в теории τ, если существует модель в $DC(\tau \cup \exists x.\phi(x))$. Таким образом, существует модель M для τ, в которой формула приобретает значение истины. Это также называется τ-выполнимостью. Запись имеет вид $M \models_\tau \phi(x)$. Формула ϕ называется валидной в теории τ, если $\forall \vec{x}.\phi(x) \in \tau$ формула истинна в любой модели M теории τ.

SAT/SMT средства предполагают, что формула задается в виде конъюнктивной нормальной формы (КНФ), элементами которой являются дизъюнкции атомарных элементов или их отрицаний. Процесс решения модели – поиск конечных значений булевых выражений и анализа конфликтов. Большая часть средств построена на алгоритме *DPLL* [1]. Для КНФ, построенных на предикатах определенной теории, существует комплексный алгоритм *DPLL(T)* [1] с решателем теорий *T*. В этом виде *DPLL* рассматривает модели, в которых

входящие без отрицания литералы могут быть атомарными высказываниями теории первого порядка.

Существует несколько применений SMT/SAT в области криптологии: верификация моделей преобразований, статистический анализ Side-channel атак, генерация тестовых случаев.

Стандарт SHA3 – конструкция на основе функций семейства Keccak-p-преобразований (permutation) [2], которые могут быть использованы в моделях различных шифрах. Некоторые инструкции раунда Keccak-p являются операциями над полиномами в $GF(2)^n$. Модель преобразования может быть построена на основе массивов слов или битовых векторов. Таким образом, для верификации могут быть применимы предикаты из теории битовых векторов (англ. *bit vector theory*) или массивов (англ. a*rray theory*) .

Задача исследования применимости SMT в моделировании и криптоанализе хеш-функций являет собой в первую очередь поиск доказательств существования возможных мест атак времени выполнения. Отдельно можно выделить применимость SMT/SAT для доказательства эквивалентности программных моделей и поиск возможных ошибок.

Применение SMT доказательства для верификации защиты преобразования Keccak-p от side-channel attack

Факультет компьютерной инженерии (англ. *ECE)* Технического Университета Вирджинии (англ. *Virginia Tech*) описывает применимость SMT на примере side-channel атак реализации криптопреобразований и метода противодействия атакам, где с секретным битом и множеством случайных битовых переменных происходит набор логических преобразований,
которые исключают возможность статистического анализа зависимости результата от секрета [3].

SMT верификация методов противодействия сводится к проверке существования зависимости критических данных от промежуточных вычислений и поиск уязвимых инструкций.

Рассмотрим математическую модель применимую для верификации метода «masking secret input» (1).

$$\begin{gathered}(x,k)\\ F(x,k)\\ I_1(x,k,r),\ldots,I_n(x,k,r)\\ \sum_{r\bullet 0,1s} I(x,k,r)\\ \sum_{r\bullet 0,1s} I(x,k',r)\\ D_{x,k},D_{x,k'}\end{gathered} \tag{1}$$

где *(x, k)* – кортеж из сообщения и ключа; *F(x, k)* – функция-преобразование; *r* – случайное значение; *I(x, k, r)* – формулы промежуточных вычислений; *s* – количество бит в r; *Σ(I(…))* – количество решений для формулы *I(x, k r)*, $D_{x,\ k}$ – вероятность *I(p, k, r)* быть логической единицей.

Тогда условие можно записать в виде выражения логики предикатов первого порядка или его отрицания:

$$\begin{gathered}\bullet\ x.\bullet\ k,k'.\left(\sum I(x,r,k)\neq\sum I(x,r',k)\right)\\ \forall\ x.\forall\ k,k'.\left(\sum I(x,r,k)=\sum I(x,r',k)\right)\end{gathered} \tag{2}$$

Выражение (1) является формулой выполнимости (англ. s*atisfiability*), когда ее отрицание – валидности (англ. *validity*). Если валидность из (2) истинно для всех значений, значит что операция *I* предельно скрыта (англ. *masked*) [3].

На практике такая запись обладает вычислительной экспоненциальной сложностью и не может быть применима.

Задача выполнимости булевых формул важна как с теоретической, так и с практической точек зрения. В теории сложности это первая задача, для которой было доказана принадлежность классу NP-полных задач. Теорема Кука утверждает, что задача о выполнимости булевой функции является NP-полной [1].

Решением на практике является использование статического анализа кода и верификации. Верификация SMT проводится для небольших размером регионов кода с применением таких подходов: для каждой инструкции производится классификация дополнительных
переменных и удалением ненужных случайных, декомпозицией и выделением атомарных операций в потоке управления, статическим анализом регионов, где безопасность гарантирована с последующим удалением их из верификации, формальной верификацией остальных частей из потока [1].

Пусть Φ определяет SMT формулу, созданную для проверки промежуточных результатов *I(x, k, r)*. Метод верификации определяет, что Φ – выполнима тогда и только тогда, когда хотя бы одна I(…) неполностью сокрыта. Определим Φ (3) как дизъюнкцию выражений исходя из (1):

$$\Phi := \left(\psi_k^{ri}\right) \wedge \left(\psi_{k'}^{ri}\right) \wedge \psi_{bti} \wedge \psi_s \wedge \psi_{diff},\ ri = \overline{0,..,2^s - 1}\ , \tag{3}$$

где составные части определены как:

$\psi_{k(')}^{r}$ – программная логика, каждая из формул кодирует копию функциональности I,

r – случайное значение взятое на промежутке ri, а ключ – k, k'. Все копии принимают одинаковое значение текста;

ψ_{bit} – Boolean-to-int кодирует преобразование булевого результата I(…) в целое число;

ψ_s – кодирует 2 суммирования логических 1 из множества результатов 2^s копий I(…);

ψ_{diff} – высказывание определяет, то что сумы должны иметь разные результаты.

Формула (3) является SMT кодированием анализа для статистической проверки зависимости результата от секретного ключа.

Для того чтобы скрытый код мог противостоять *DPA* – атакам первого порядка (англ. *First-order DPA*) [4], все подпрограммы *I(x, r, k)* должны быть скрыты. Тем не менее это свойство эффективно при *DPA* атаках высших порядков (англ. *High-order DPA*) [4], где наблюдатель может считывать значения сразу с нескольких промежуточных результатов.

Таким образом можно записать условие выполнимости защиты против *high-order DPA:*

$$\bullet\ x\ .\bullet\ k,k'\ .\left(\sum_{r\bullet\ 0,1}\sum_{i=0}^{d} I_i(x,k,r) \neq \sum_{r\bullet\ 0,1}\sum_{i=0}^{d} I_i(x,k,r)\right) \tag{4}$$

Для эксперимента были выбрана функции *Keccak-p*, реализация которой взята из официального репозитория разработчиков. Таблица хранит условия и результат работы алгоритма Sleuth [3] для различных вариантов:

P1 – Mac-keccak 512 masked;

P2 – Masked Chi step from Keccak-p;

P3 – Mac-keccak 512 not-perfect mask in Chi.

Таблица содержит усредненные результаты проведенных испытаний нахождения регионов кода, которые не полностью защищены. Реализация взята из NIST Keccak Reference [2].

P	Nodes	Plains	Keys	Rands	Masked	Nodes failed	Nodes checked	Time
P1	128k	288	288	805	T	0	128k	92m

P	Nodes	Plains	Keys	Rands	Masked	Nodes failed	Nodes checked	Time
P2	19	3	0	4	F	1	19	0.15s
P3	128	288	288	805	F	512	128k	113m

Поиск минимальных диференциальных характеристик с помощью CryptoSMT

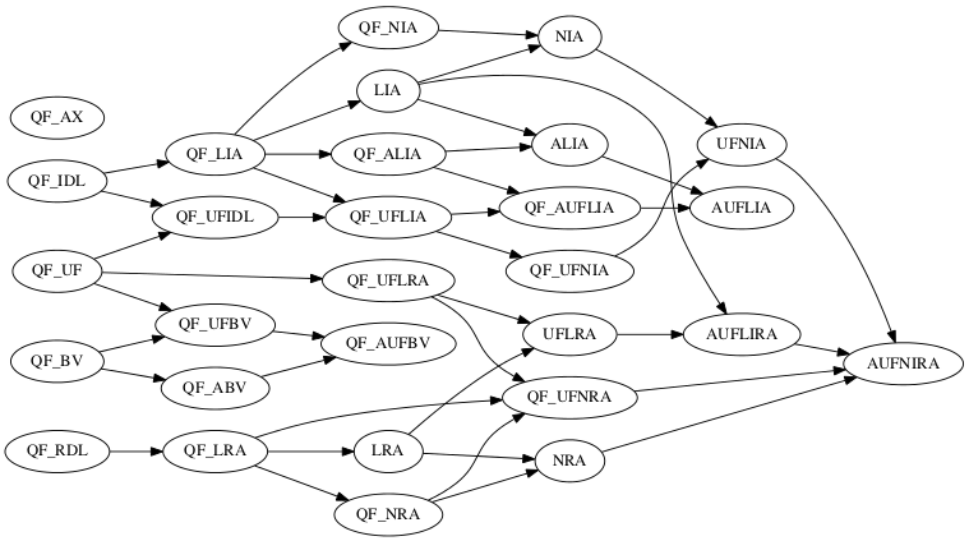
Криптоанализ хеш-преобразований обычно включает себя дифференциальный криптоанализ и поиск коллизий, обратных отражений функций.

Семейство преобразований Keccak-p основано на операции над массивами бит, таким образом КНФ должна быть основана на высказываниях теории битовых векторов.

CryptoSMT – инструмент, построенный на базе SMT доказательств. Для работы утилиты необходимы STP, CryptoMiniSat, Boolector решатели. Работа этих инструментов основана на стандарте SMT-lib 2.0, который содержит требования к решателю теорий [5].

Стандарт SMT-lib 2.0 опредеяет теорию как множество высказываний и операций элементов теории.

На рисунке изображена иерархия теорий, представленная в SMT-lib 2.0 [6].



Применение SAT доказательств для дифференциального анализа преобразований подразумевает: реализацию КНФ модели преобразования, формальной верификации преобразований, КНФ поиска минимального веса дифференциальной характеристики.

Рассмотрим в качестве примера формирование теоремы модели линейного преобразования в раунде, основанной на атомах теории битовых векторов из SMT-LIB 2.0. [6]

```
  for i in range(5):
  command += "ASSERT({} = BVXOR({}, BVXOR({}, BVXOR({}, BVXOR({},
{})))));\n".format(
      c[i + 5*rnd], s[i + 5*0 + 25*rnd], s[i + 5*1 + 25*rnd],
      s[i + 5*2 + 25*rnd], s[i + 5*3 + 25*rnd], s[i + 5*4 + 25*rnd])
```

Формирование теоремы вычислений веса дифференциальной характеристики можно привести как:

```
    command += ("ASSERT({0} = (IF {1} = 0b{4} THEN BVSUB({5},0b{4},0b{6}1)"
"ELSE BVXOR({2}, {3}) ENDIF));\n".format(
    mp[z + wordsize*y + 5*wordsize*rnd],
    xin[z + wordsize*y + 5*wordsize*rnd],
    varibits, doublebits, "1"*5,
    5, "0"*4))
```

Среднее время поиска характеристик на основе сконструированной теоремы преобразований (Intel Core i7 2.2Hz, 4Gb RAM) – 113min.

Проверка эквивалентности функций с помщью Cryptol

Язык Cryptol является функциональным DSL *(Domain Specific Language*) языком, с возможностью верификации описанной модели посредством теорем. В Cryptol 2.0 основным решателем является Z3 от Microsoft Research.

Следующая запись объявляет теорему для проверки эквивалентности двух аргументов. Тип возвращаемого значения – всегда логическое значение в виде бита.

```
    T: [8] → [8]→ [Bit]
    theorem T: {x, y}. x == y.
```

Cryptol предоставляет несколько инструментов для работы с теоремами. Quickcheck – технология, появившаяся для языка Haskell, идеей которой является генерация случайных тестов, основанных на определении теоремы с мономорфными, конечными типами аргументов. Так команда *«:check»* возвращает не только результат, но и покрытие кода тестами.

В случае когда результат теоремы – FALSE, интерпретатор возвращает набор значений, в которых не выполняется условие. Команда *«:exhaust»* проводит тестирование для всех возможных наборов аргументов теоремы. [7]

Формальные доказательства теорем (*«:prove»*) реализованы с помощью проверки на эквивалентность функций, где вторая функция – логическая ИСТИНА.

Модульное тестирование Keccak-p можно определить как множество теорем, проверяющих эквивалентность преобразования битового сообщения с валидным результатом из технической документации. Следующий код содержит пример использования доказательств теорем для тестирования.

```
    theorem testsPass: tests == ~zero;
    tests = [t00 t01 t02 t03 ...];
    t01 = SHA_3_224 (r `{n=0} 0x00) == md
0xF71837502BA8E10837BDD8D365ADB85591895602FC552B48B7390ABD;
    t02 = SHA_3_224 (r `{n=1} 0x00) == md
0x860E3EC314C5CBF19C1A4314E9EA8CB85CECD18BD850B42F5C6F2A07;
    ...
    t40 = Keccak_r544c256 (r `{n=0} 0x00) == md
0xA3CEA55CFD9F4432AD3F9AE33673AE12665F66D150A11AF54E007C7F26F7C9A6E69862E14A2BAD
40048D439E26FB67B40807412BAE2EB42B6896B1D4D602755B23EC19E4;
    t41 = Keccak_r544c256 (r `{n=70} 0xF8CB65B7FE6995F200) == md
0x730AD05FCC3DD4E250D00D5426A42BDAEB8615772F1D1BC0F9AE2639F8920A1BF36CE60F893EAF
E782095A903848C7B150E79B3AAB32C1C3EDE8AB4E64D4015E5E47021B;
```

Выводы

Изучены варианты применения SMT/SAT в криптологии, где верификация осуществляется с помощью логических выражений и является формальным методом. Был проведен эксперимент доказательства достаточной защиты шагов Keccak от side-channel атак.

CryptoSMT является ярким представителем инструментов, использующих SMT (SAT) для криптоанализа преобразований, с применением решателей теорий первого порядка.

Поиск обратных отражений раунда Keccak-p можно произвести с помощью теории битовых векторов.

Язык Cryptol использует Z3 prover для проверки эквивалентности функций и формальной верификации на основе теорем. Теоремы могут быть использованы для генерации и тестирования частей модели в случае мономорфности и конечности аргументов. Верификация модели основана на технологии эквивалентности.

Список литературы: 1. *Nieuwenhuis, R., Oliveras, A., Tinelli, C.* Solving SAT and SAT Modulo Theories: From an Abstract Davis-Putnam-Logemann-Loveland Procedure to DPLL(T) // Journal of the ACM. – 2006. – Т. 53, № 6. – P. 937 – 977. 2. *Guido Bertoni, Joan Daemen, Michaël Peeters Gilles Van Assche.* The Keccak sponge function family [Электронный ресурс]. – Режим доступа: http://keccak.noekeon.org/ – 21.04.2017. – Загл. с экрана. 3. *Hassan Eldib, Chao Wang, and Patrick Schaumont* SMT-Based Verification of Software Countermeasures against Side-Channel Attacks [Электронный ресурс]. – Режим доступа: http://www.faculty.ece.vt.edu/chaowang/pubDOC/Eldib14maskVerif.pdf – 21.02.2011. – Загл. с экрана. 4. *Брюс Шнайер.* Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке С. – Москва : Триумф, 2002. 5. *Clark Barrett, Aaron Stump, Cesare Tinelli.* The SMT-LIB Standard Version 2.0 [Электронный ресурс]. – Режим доступа: http://smtlib.cs.uiowa.edu/papers/smt-lib-reference-v2.0-r10.12.21.pdf – 21.11.2011. – Загл. с экрана. 6. *SMT-LIB* The satisfiability modulo theories library [Электронный ресурс]. – Режим доступа: http://smtlib.cs.uiowa.edu/logics.shtml – 21.03.2017. – Загл. с экрана. 7. *Levent Erk* ok Theorem declarations in Cryptol [Электронный ресурс]. – Режим доступа: http://community.galois.com/pipermail/cryptol-users/attachments/20110316/f66e99b2/attachment-0003.pdf – 02.10.2008. – Загл. с экрана.

Харьковский национальный
университет радиоэлектроники

Поступила в редколлегию 04.04.2017