Практические оценки сложности регулярных выражений

Д. Е. Александров[1]
А. В. Красненкова[2]

Эта работа описывает серию экспериментов над конечными автоматами и выявляет классы потенциально простых и потенциально сложных регулярных выражений, ведущих к экспоненциальному взрыву в числе состояний.

Ключевые слова: конечные автоматы, регулярные выражения, экспоненциальный взрыв, вычислительные эксперименты

1. Введение

Проблема выявления вторжений является актуальной проблемой компьютерной безопасности: с каждым годом число разнообразных атак только растет [1]. Одним из самых распространенных подходов в выявлении вторжений является сигнатурный подход, в рамках которого экспертные знания о существующих атаках и уязвимостях формализуются в терминах регулярных выражений; проверка соответствия текущего состояния системы одной из известных сигнатур осуществляется с помощью различных модификаций конечных автоматов [2]. В случае использования конечных детерминированных автоматов (ДКА) временная сложность распознавания линейна по длине входного слова, однако число состояний может экспоненциально зависеть от длины выявляемых регулярных выражений; переход к недетерминированным конечным автоматам (НКА) снижает нагрузку на память до линейной, однако ухудшает временную сложность. Известен ряд конструкций, позволяющих в некоторых случаях снизить пространственную сложность ДКА (см., например, обзор [2]), однако такие конструкции заведомо хороши для некоторых классов регулярных выражений, а утверждения о работе в

[1] *Александров Дмитрий Евгеньевич* — к.ф.-м.н., доцент каф. вычислительной математики мех.-мат. ф-та МГУ, e-mail: d06alexandrov@gmail.com

Alexandrov Dmitriy Evgenievich— Ph.D., Associate Professor, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Computational Mathematics.

[2] *Красненкова Анастасия Владимировна* — аспирант каф. МаТИС мех.-мат. ф-та МГУ, e-mail: ankrasnenkova@gmail.com

Krasnenkova Anastasiya Vladimirovna — graduate student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

случае произвольных выражений отсутствуют. Также имеются исследования о выделении классов регулярных выражений, эффективно распознаваемых НКА (см., например, [3]).

В случае аппаратной реализации сигнатурной компоненты системы выявления вторжений представляется разумным основываться на ДКА, так как система критична к скорости работы. Разумной стратегией исследования представляется выделение практически интересных классов регулярных выражений, приводящих к ДКА с большим числом состояний (например, с помощью анализа известных баз сигнатур, таких как база системы snort [4]), и решение проблемы для найденных классов.

2. Система для анализа автоматной сложности

Для проведения вычислительных экспериментов был использован программный комплекс, позволяющий строить недетерминированные и детерминированные конечные автоматы приведенного вида для распознавания как одиночных регулярных выражений, заданных в нотации PCRE, так и их объединений. Необходимые определения можно найти, например, в работе [5], описание формата PCRE — в [6].

Для вывода на экран числа состояний детерминированных конечных автоматов приведенного вида, каждый из которых распознает по одному из PCRE-совместимых регулярных выражений, записанных в файле, используется следующая команда:

```
./re2fa --input-type=regexp-file -v -m expressions_example
```

где `expressions_example` — путь к файлу с выражениями.

С целью анализа множества регулярных выражений на предмет экспоненциального взрыва числа состояний используется следующая команда, позволяющая найти число состояний ДКА, распознающего объединение регулярных выражений:

```
./re2fa --input-type=regexp-file -v -m -j expressions_example
```

где `expressions_example` — путь к файлу с выражениями, которые необходимо объединить.

Приведенные ниже эксперименты могут быть воспроизведены, используя код и тестовые примеры, расположенный по ссылке https://github.com/d06alexandrov/re2fa.

3. Вычислительные эксперименты

Проведем серию вычислительных экспериментов. Первая серия экспериментов направлена на исследование примеров регулярных выражений,

которые, имея невысокую сложность составных частей, дают невысокую сложность итоговых выражений. Вторая серия экспериментов моделирует прямо противоположную ситуацию: выражения с невысокой автоматной сложностью составных частей при объединении дают резкий сложностной скачок.

3.1. Работа с простыми классами регулярных выражений

В качестве примеров регулярных выражений, имеющих низкую автоматную сложность и невысокую временную сложность, рассмотрим следующие шаблоны регулярных выражений из работы [3]:

№	1	2	3	4	5	6
тип	(ac|bc)+	(a+|b+|c+)+	(a|ab|bc)+	(a+b+c+)+	(a+b+)+|(b+c+)+	(abc)+|(bc)+

Эти выражения оцениваются как имеющие линейную или субквадратичную временную сложность распознавания с точки зрения НКА [3].

Зафиксируем значения параметров a, b, c, подставив вместо них выражения из базы snort:

```
a = /text3Gtrack*?textBox*?(xy)\s*\x3D*(\x22\x27\s)*\x2d\d/
b = /textBox*?(xy)\s*\x3D(\x22\x27\s)+\d{5}/
c = /endobjhttp\x3A\x2F\x2F*?\x2EloadXML/
```

Сложность данных регулярных выражений в терминах ДКА составляет 25, 18 и 21 состояний.

В результате экспериментов получили следующие сложностные результаты (в терминах ДКА):

номер шаблона	1	2	3	4	5	6
сложность	60	59	61	62	78	38

Полученные экспериментальные результаты позволяют сформулировать гипотезу о том, что идентифицированные в [3] классы, являющиеся “простыми” с точки зрения НКА, просты и с точки зрения ДКА.

3.2. Поиск “сложных” классов регулярных выражений

Рассмотрим следующий набор регулярных выражений из базы системы snort, позволяющий выявлять вредоносные воздействия на pdf-файлы (обфускация, DoS-атаки и другие) и некоторые иные атаки:

```
/<script.*?(&#\d+\x3b){30}/si
/Collab\x2EaddStateModel\s*\x28\s*\x7B.*cName\s*\x3A\s*\x22(\x22|\x5Cx00)/si
/JBIG2Decode.*?stream(\x0d\x0a|\x0a|\x0d)/si
/Type\s*\x2FAction.*?Launch.*?\x28\s*\x2f\w/si
/<\x21ENTITY[^>]+SYSTEM[^>]+http\x3A\x2F\x2F.*?\x2EloadXML/si
/obj\x3c\x3c.*?\x2fBaseFont\x2f[^\x80-\xff\x2f]*[\x80-\xff].*?endobj/s
/textEncoding=\x22.*&#x0?D\x3b&#x0?A\x3b/i
/doctype\s+svg\s+.*?\x28\s*?%pipe%[^\x29]*?[\x60\x3b\x7c\x23]/is
/^Rar\x21\x1A\x07\x00.*\x2E(cmd|bat|com|exe|scr|pif)/si
/^static\s+(\w+\s+)?char\s*\x2A\s*\w+\s*\x5B\x5D\s*\x3D\s*\x7B.*\x22[^\x22]{200}/si
/text3Gtrack.*?textBox.*?[xy]\s*\x3D*[\x22\x27\s]*\x2d\d/si
```

```
/textBox.*?[xy]\s*\x3D[\x22\x27\s]+\d{5}/si
/\x3C\x21ENTITY\s+.*\s+[\x22\x27]\x5D\x3E.*\x3Cd.*[\x22\x27]\x3E/s
```

Если мы будем рассматривать эти выражения по отдельности, то автоматы для них имеют небольшое количество состояний: 128, 34, 19, 21, 32, 32, 27, 21, 22, 228, 24, 16, 18 соответственно. При последовательном их объединении мы получим: 317 $\to$ 697 $\to$ 2229 $\to$ 9549 $\to$ 41081 $\to$ 91249 $\to$ 237585 $\to$ 503400 состояний (затем процесс был остановлен системой из-за исчерпания памяти), что позволяет классифицировать данную ситуацию как экспоненциальный взрыв.

4. Заключение

Имеющийся у нас программный комплекс позволяет отделять простые (в автоматном плане) регулярные выражения от сложных. Наши дальнейшие планы таковы: добавить в программу известные методы понижения сложности (см. [2]), расширить поддержку PCRE-нотации и постараться выявить новые классы сложных регулярных выражений.

Список литературы

[1] Галатенко А. В., "Активный аудит", *JetInfo*, 1999, № 8, 3–28.

[2] Александров Д. Е., "Эффективные методы реализации проверки содержания сетевых пакетов регулярными выражениями", *Интеллектуальные системы. Теория и приложения*, **18**:1 (2014), 37–60.

[3] Backurs A., Indyk P., "Which regular expression patterns are hard to match?", Proceedings of the 2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS), 2016, 457–466.

[4] https://snort.org/, страница системы выявления вторжений snort.

[5] Кудрявцев В. Б., Алешин С. В., Подколзин А. С., *Введение в теорию автоматов*, Наука, 1985, 320 с.

[6] https://www.pcre.org/original/doc/html/pcrepattern.html, страница документации нотации PCRE.

Practical estimates of the complexity of regular expressions

D.E. Alexandrov, A.V. Krasnenkova

This paper describes a series of experiments on finite automata and identifies classes of potentially simple and potentially complex regular expressions leading to an exponential explosion in the number of states.

Keywords: finite automata, regular expressions, exponential explosion, computational experiments

References

[1] Galatenko A. V., "Active audit", *JetInfo*, 1999, №8, 3–28 (In Russian).

[2] Alexandrov D. E., "Effective methods of implementation of checking the contents of network packets with regular expressions", *Intellectual Systems. Theory and Application.*, **18**:1 (2014), 37–60 (In Russian).

[3] Backurs A., Indyk P., "Which regular expression patterns are hard to match?", Proceedings of the 2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS), 2016, 457–466.

[4] https://snort.org/, page of network intrusion detection system named snort.

[5] Kudryavtsev V. B., Alyoshin S. V., Podkolzin A. S., "Introduction To Automata Theory", 1985 (In Russian).

[6] https://www.pcre.org/original/doc/html/pcrepattern.html, PCRE notation documentation page.