электронный журнал
МОЛОДЕЖНЫЙ НАУЧНО-ТЕХНИЧЕСКИЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51038.

УДК 004.415.53

Эффективное тестирование реализации сетевого протокола при полном обходе конечного автомата

Д.Б. Агеев, *студент*
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Программное обеспечение ЭВМ и информационные технологии»

Научный руководитель: В. А. Крищенко, к.т.н, доцент
Россия, 105005, г. Москва, МГТУ им. Н.Э. Баумана,
кафедра «Программное обеспечение ЭВМ и информационные технологии»
irudakov@bmstu.ru

Тестирование сетевой службы должно включать системные тесты, созданные на основе спецификации протокола, который реализует служба. Программная реализация службы при этом может рассматриваться как некоторый «черный ящик», который при получении сообщения клиента посылает ему ответ и меняет своё состояние некоторым образом. При этом смена состояний должна соответствовать конечному автомату серверной стороны реализуемого протокола.

При системном тестировании точное состояние тестируемой процесса сетевой службы в произвольный момент времени неизвестно, поэтому единственным способом вернуться к предыдущему состоянию при обходе графа является сброс состояния в начальное состояние и повторная посылка последовательности команд, за исключением последней. При этом единственный вывод о состоянии сетевой службы можно сделать по ответу сервера, который проверяется в ходе выполнения теста. Это не позволяет применить для тестирования такой службы методы тестирования «чистых» конечных автоматов, описанные, например, в [1].

В рамках данной работы предлагается метод тестирования сетевой службы с использованием системного вызова клонирования (POSIX-функция **fork**, [2]), который позволяет копировать полный контекст родительского процесса. Копирование родительского контекста позволяет избавиться от необходимости постоянного сброса состояния сетевой службы, что позволяет сократить число переданных команд, необходимых для полного обхода графа состояний протокола. Явным недостатком такого подхода являются затраты на копирование контекста родительского процесса, поэтому необходимо проведение эксперимента с программной реализацией данного подхода: при

наличии в операционной системе механизма копирования памяти клона при записи в неё предлагаемый подход может оказаться успешным.

Состояния сетевого протокола можно представить в виде конечного автомата $M = (Q, V, q_0, \delta, F)$, где Q — множество состояний протокола сервера, V — множество допустимых команд, которые переводят протокол сервера из одного состояния в другое, q_0 —состояние протокола сервера, δ — множество правил перехода из одного состояния протокола сервера в другое, F — множество заключительных состояний протокола сервера.

Сервер должен обрабатывать тем или иным образом любую существующую в протоколе команду в каждом своём состоянии, поэтому из каждого состояния осуществляется переход по всем возможным командам. В связи с этим для экспериментов с реализованным методом тестирования была выбрана модель состояний в виде n-полного дерева, в которой рассматриваются абсолютно все возможные комбинации команд длины h: по всем парам (q, v), где $q \in Q$, $v \in V$ существует правило перехода. Высота такого такого дерева на единицу больше длины максимальной последовательности команд, посылаемой серверу.

Проблема тестирования сетевой службы заключается в том, что невозможно гарантировано вернуться к предыдущему состоянию службы. Поэтому со стороны клиента отправляются все возможные последовательности команд, в которых количество команд равно высоте n-арного дерева конечного автомата, при этом после каждой уникальной последовательности происходит возврат сервера к начальному состоянию, что неэффективно и приводит к значительному увеличению времени тестирования (рис. 1). При использовании системного вызовы fork дочернего процесса родительский процесс сможет принять новые команды от клиента без возврата состояния протокола сервера в начальное (рис. 2).

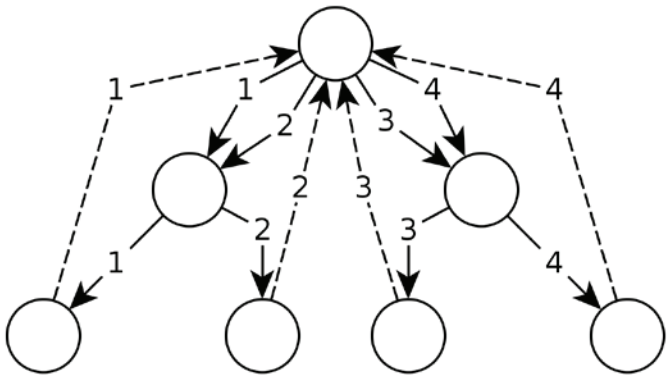

Рис. 1. Обход дерева без использования клонирования

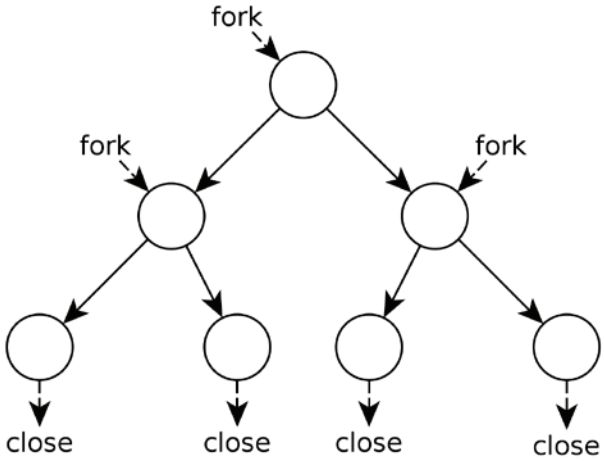

Рис. 2. Обход дерева при использовании клонирования

Без использования клонирования процессов количество команд, которые необходимо отправить серверу, равно $k = n^h h$, а с использованием клонирования —

$$k = \sum_{i=1}^{h} n^i$$

где n — количество всевозможных команд, h — максимально возможная длина последовательности команд, посылаемая серверу. Для всех натуральных значений n и h верно

$$\sum_{i=1}^{h} n^i \leq n^h h$$

Сокращение количества команд может привести к уменьшению времени тестирования, если экономия на их

передачу и обработку превысит накладные расходы на использование системного вызова клонирования. Использование метода без копирования контекста требует многократной передачи одних и тех же команд при тестировании, поэтому, чем больше время передачи и обработки одной команды, тем более эффективен метод с использованием системного вызова fork. Целесообразно проанализировать зависимость времени тестирования от размера передаваемого между клиентом и службой сообщения. Для этого необходимы алгоритмы генерации данных на стороне клиентов, а также алгоритмы обработки полученных данных со стороны сервера. Алгоритмы на стороне клиента для двух рассматриваемых методах также как и для алгоритмов обработки на стороне сервера, отличаются так, как в случае использования клонирования необходимо передавать данные в определенном порядке в соответствии с выбранным правилом обхода дерева, а в случае без клонирования генерируются все возможные последовательности команд. Ниже приведены алгоритмы работы тестируемой сетевой службы без использования и с использованием системного вызова fork.

Листинг 1. Работа службы при тестировании без использования клонирования

```
повторить:
    получить  команду текущая_команда
    перейти в новое состояние по команде текущая_команда
    текущая_глубина = текущая_глубина + 1
    если текущая_глубина >= максимальная_глубина:
        текущая_глубина = 0
        сброс состояния протокола сервера
    подготовить ответ для отправки
    отправить ответ
```

Листинг 2. Работа службы при тестировании без использованием клонирования

```
повторить:
    получить команду текущая_команда
    создать новый процесс с идентификатором ид_потомка
    если ид_потомка == 0:
```

```
перейти в новое состояние по команде текущая_команда
текущая_глубина = текущая_глубина + 1
текущий_потомок = 0
отправить ответ
если текущая_глубина >= максимальная_глубина:
   завершить процесс
иначе:
  ожидание завершения процесса с идентификатором ид_потомка
   текущий_потомок = текущий_потомок + 1
   если текущий_потомок >= количество_потомков и текущая_глубина != 0:
   текущий_потомок = 0;
   завершить процесс;
```

Для временного анализа методов тестирования сервера с использованием и без использования системного вызова fork была написана программа на языке C++ в соответствии с описанными выше алгоритмами. Вычислительный эксперимент охватывал высоту дерева 5, 10 и 15. На рис. 3 приведено зависимость времени тестирования от размера передаваемой на сервер команды при $h = 5$ для обоих рассмотренных подходов. На рис. 4 приведена аналогичная зависимость для $h = 10$, на рис. 5 — для $h = 15$.

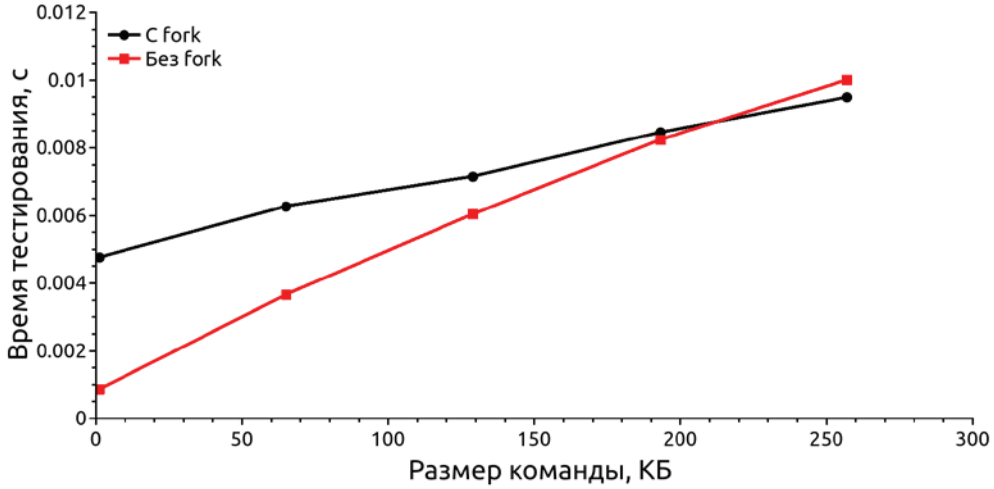

Рис. 3. Зависимость времени тестирования от длины команды при $h = 5$

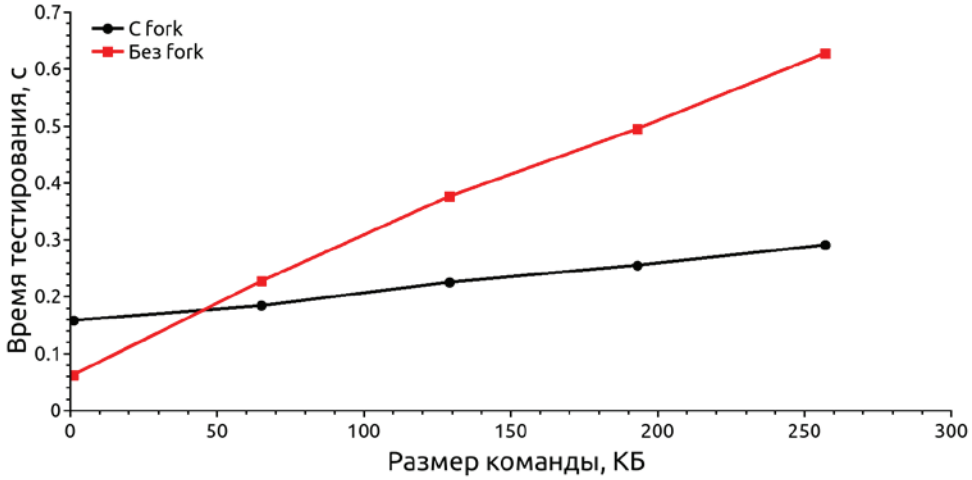

Рис. 4. Зависимость времени тестирования от длины команды при $h = 10$

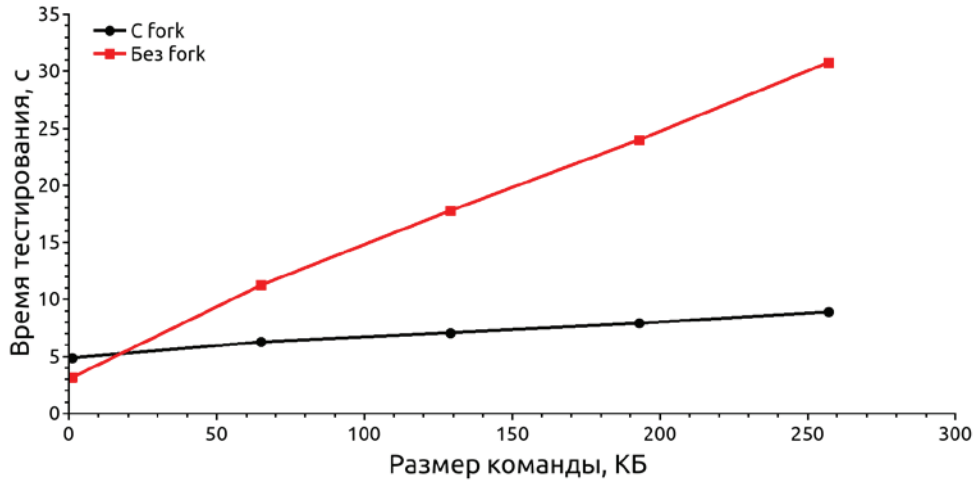

Рис. 5. Зависимость времени тестирования от длины команды при $h = 15$

Из проведенного эксперимента видно, что метод без использования клонирования дает лучший результат при небольшом размере передаваемой команды, а при увеличении её размера предлагаемый метод начинает давать выигрыш во времени тестирования, причём выигрыш становится всё больше с ростом тестируемого автомата.

Список литературы

1. D. Lee, M. Yannakakis. Principles And Methods Of Testing Finite State Machines - A Survey. Proceedings of the IEEE , 1996, Vol. 84, Issue 4, pp. 1090-1123.
2. У. Ричард Стивенс, Стивен А. Раго. UNIX. Профессиональное программирование, 2-ое издание. М: Символ-Плюс, 2007 — 1040 с.