УДК 004.451.2

Трещев Иван Андреевич, кандидат технических наук, доцент кафедры «Информационная безопасность автоматизированных систем», Комсомольский-на-Амуре государственный университет
Treshchev Ivan Andreevich, Candidate of Engineering Sciences, Associate Professor of the Department "Information Security of Automated Systems", Komsomolsk-on-Amur State University
Кудряшова Екатерина Сергеевна, кандидат физико-математических наук, доцент кафедры «Проектирование, управление и разработка информационных систем», Комсомольский-на-Амуре государственный университет
Kudryashova Ekaterina Sergeevna, Candidate of Physical and Mathematical Sciences, Associate Professor of the Department "Design, Management and Development of Information Systems", Komsomolsk-on-Amur State University
Ватолина Анастасия Сергеевна, студент, Комсомольский-на-Амуре государственный университет
Vatolina Anastasia Sergeevna, student of Komsomolsk-na-Amure State University

МЕТОДЫ ОБНАРУЖЕНИЯ И ПРЕДУПРЕЖДЕНИЯ СОСТОЯНИЯ «ГОНКИ»

METHODS FOR DETECTING AND WARNING THE "RACE" STATE

Аннотация. Статья посвящена методам обнаружения состояния «гонки». Приведены признаки наличия уязвимостей, связанных с рассматриваемым состоянием. Приведена сеть Петри для расслоенных критических секций описывающая «гонки». Известно, что использование семафоров или других объектов синхронизации не решит проблемы, возникающих при взаимном исключении. Приведен пример, когда два процесса, имеющие критические секции, соответствующие двум разным ресурсам, исполняются параллельно и возможно перейдут во взаимную блокировку. Отмечено, что операционные системы Windows и Linux подвержены риску возникновения состояния гонки. Корректная разработка с учетом возможности возникновения тупиков и использование отладчиков позволяет значительно уменьшить вероятность возникновения блокировок.
Abstract. The article is devoted to methods for detecting the state of "race". The signs of the presence of vulnerabilities associated with the state under consideration are given. A Petri net for layered critical sections describing "races" is given. It is known that the use of semaphores or other synchronization objects will not solve the problems that arise from mutual exclusion. An example is given where two processes that have critical sections corresponding to two different resources are executed in parallel and possibly become deadlocked. It has been noted that Windows and Linux operating systems are at risk of a race condition. Correct development taking into account the possibility of deadlocks and the use of debuggers can significantly reduce the likelihood of blocking

Ключевые слова: методы, обнаружение, состояние «гонки», параллельное программирование, уязвимость.
Key words: methods, detection, "race" state, parallel programming, vulnerability.

Введение

«Гонки» - ошибка, часто возникающая при разработке многопоточных систем, или приложений [1]. В этом состоянии конечный результат зависит от того, в каком порядке выполняются участки кода, точнее результат работы алгоритма теряет свою детерминированность и на одних и тех же наборах данных мы можем получать различные результаты, поэтому более верно использовать термин неопределённость паралле-

лизма [2]. Отметим, что возможна полная блокировка функционирования разрабатываемого приложения и переход его в «zombie» состояние [4]. Состояние гонки – непостоянная ошибка, проявляющаяся в случайные моменты времени и «исчезающая» при попытке её локализовать или отладить [5].

В системах с параллелизмом существуют временные задержки связанные с необходимостью синхронизации или обмена сообщениями между вычислительными узлами, а в случае многопоточных приложений аналогичная ситуация наблюдается и для процессов [3].

Уязвимости

Основные признаки эксплуатации уязвимостей, связанных с использованием состояния гонки:

- несанкционированный доступ объекту;
- нарушение целостности объекта;
- нарушение доступности.

Характерные свойства, обуславливающие наличие ошибки:

- совместно используемый объект;
- конкурирующие потоки или процессы исполнения, либо же дочерние потоки некоторого родителя;
- изменение совместно используемого объекта одним из участников гонок.

Статистический и динамический анализ кода используют для обнаружения наличия состояния гонки в разрабатываемых приложениях.

Уязвимости, связанные с состоянием гонки существовали в следующих версиях программного обеспечения:

- Firefox 2007;
- Internet Explorer 2011;
- «ярлык» Windows;
- COW - «копирование при записи»;
- setuid-root/usr/bin/at;
- гонки Ptrace в ядре Linux.

Способы предотвращения ошибки «Состояние гонки»

Типичное решением для состояния гонки является обеспечение того, приложение имеет ограниченные права на общие данные, пока один поток или процесс манипулирует получением особых прав на доступ к данным, другие ожидают. Такой процесс называется запиранием. В качестве механизмов, используемых для запирания, наиболее часто используются:

- Семафоры.
- Мьютексы.
- Блокирование файлов.
- Конвейеры.

Рассмотрим пример, возникающий при использовании критических секций, иллюстрирующий расслоение. Использование семафоров не решает проблемы, возникающих при взаимном исключении. Рассмотрим два процесса P_1 и P_2, имеющих критические секции, соответствующие двум разным ресурсам. Пусть процесс P_1 содержит операторы, которые захватывают ресурс M, затем M*, освобождая их в обратном порядке T(M);…;T(M*);…;G(M*);…;G(M), а процесс P_2 операторы T(M*);…; T(M);…; G(M);…; G(M*), наоборот захватывающий M*, а затем M при этом освобождение ресурсов так же производится в обратном порядке. Иллюстрация приведена на рисунке 1. Ясно, что области A и B являются потенциально опасными и, вообще говоря, могут содержать уязвимости, связанные с состоянием «гонки».

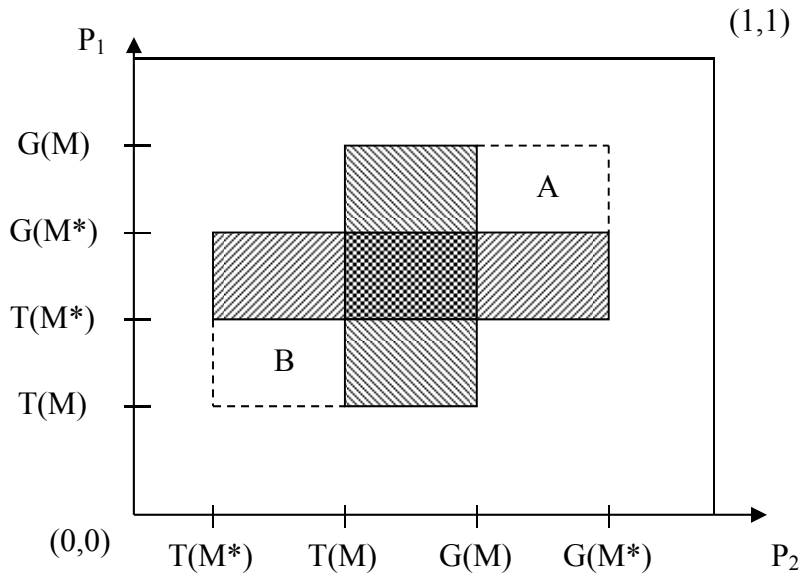

Рисунок 1 – Иллюстрация расслоения критических секций

Точки плоскости соответствуют всем парам состояний процессов. Закрашенные области - недоступные пары состояний. Отметим, что если процессы после запуска через некоторое время окажутся в состоянии, соответствующем точке области B, то они будут заблокированы на бесконечное время, поскольку каждый из них будет ожидать соответствующего ресурса, который блокируется вторым.

Заключение

В данной работе рассмотрены уязвимости, приводящие к состоянию гонки. Стоит отметить, что различные версии операционных систем Windows и Linux по-прежнему подвержены риску возникновения состояния гонки, например уязвимость ядра Linux или Dirty COW. Это указывает на то, что даже после исследований в этой области существует разрыв между существующими методами обнаружения и предотвращения состояния гонки. Также в работе были рассмотрены способы и методы обнаружения ошибки, а именно статическое и динамическое тестирование. Использование отладчиков позволяет значительно уменьшить вероятность появления состояния гонки в готовом продукте. Описаны способы предотвращения и устранения ошибки состояние гонки в программных продуктах.

СПИСОК ИСТОЧНИКОВ

1. Кудрин, М. Метод нахождения состояний гонки в потоках, работающих на разделяемой памяти / М. Кудрин, А. Прокопенко, А. Тормасов // Труды МФТИ. Том 1, № 4, 2009. С. 182-201.

2. Одинцов, И. Профессиональное программирование. Системный подход. / И. Одинцов. – СПб : БХВ-Петербург, 2004. – 624 с.

3. Салищев, С. И. Использование языков и сред управляемого исполнения для системного программирования / С. И. Салищев, Д. С. Ушаков // Системное программирование. – 2009. – Т. 4 № 1 - С. 198-216.

4. Сергей, И. Д. Реализация гибридных типов владения в Java посредством атрибутных грамматик / И. Д. Сергей // Системное программирование. – 2011.-Т.6.–№1– С. 49-79.

5. Трифанов, В. Ю. Динамическое обнаружение состояний гонки в многопоточных Java-программах / В. Ю. Трифанов // Математико-механический факультет СПБГУ. – 2013. – URL: http://www.math.spbu.ru/user/dkoznov/papers/vtrifanov.pdf (дата обращения: 12.11.2015).