Мониторинг потока управления процессов в операционных системах на основе графов потока вызовов

Пучкин Данила Андреевич, Danila.Puchkin@kaspersky.com
АО «Лаборатория Касперского», Россия, Москва, 125212, Ленинградское шоссе, д.39А, стр.3, БЦ «Олимпия Парк»

1 Постановка задачи

Под мониторингом потока управления процесса в операционной системе понимается сравнение фактического потока управления процесса с некоторой эталонной моделью. В работе [1] предложен подход к контролю корректности исполнения процессов, основанный на мониторинге потока управления. Под корректным исполнением процесса понимается функционирование процесса в соответствии с предъявляемыми ему требованиями.

В рассматриваемом подходе предлагается контролировать поток управления в объеме исполняемых процессом целевых команд на основе некоторой эталонной модели. Целевые команды (целевые инструкции) – это команды, информация об исполнении процессом которых доступна от операционной системы, в рамках которой функционирует процесс. В качестве эталонной модели используется сигнатура потока управления – модель процесса, определяющая последовательности целевых инструкций, которые могут быть исполнены данным процессом. Конкретный вид модели, как и конкретные алгоритмы мониторинга потока управления в рассмотренной работе не уточняются.

В работе предлагается способ мониторинга потока управления процессов в операционных системах, основанный на использовании одной из форм сигнатуры потока управления. Работа ограничивается случаем, когда контролируемый процесс является однопоточным, во время его исполнения не происходит системных прерываний, а его программа не содержит команд переходов, кроме вызовов функций.

2 Сигнатура потока управления

Сигнатура потока управления процесса генерируется из набора графов потока управления функций программы данного процесса. В ходе генерации сигнатуры из вершин базовых блоков графов потока управления исключаются все инструкции, не являющиеся целевыми или управляющими командами. Управляющие команды – это команды, определяющие поток управления. Примерами таких команд являются команды вызова функций, команды перехода, команды установки программных исключений потока, команды создания потоков и другие. В случае, если в базовом блоке нет целевых или управляющих инструкций, из вершины такого базового блока исключается вся информация о командах.

После удаления из вершин базовых блоков всех команд, кроме целевых или управляющих инструкций, каждая вершина, содержащая более одной инструкции, заменяется на путь. В данном пути каждой вершине соответствует одна команда из преобразованного базового блока. Вершины в данном пути следуют в порядке, соответствующем инструкциям заменяемой вершины. При замене все ребра, входящие в заменяемую вершину, будут входить в первую вершину данного пути, а выходящие из нее – в последнюю. Если вершине преобразованного базового блока соответствует одна команда или не соответствует ни одной команды, ее не требуется преобразовывать.

В результате преобразованный таким образом граф будем содержать в себе вершины, или соответствующие одной инструкции, или не соответствующие ни одной. Вершины, не соответствующие ни одной команде, будем называть пустыми, а остальные вершины – непустыми. Вершины, соответствующие целевой или управляющей команде, будем называть целевыми или управляющими вершинами соответственно.

Получающийся в результате граф называется графом потока вызовов – это граф потока управления, в котором есть входная и выходная вершины, а каждой непустой вершине соответствует одна целевая или управляющая команда. Каждая вершина графа потока вызовов имеет ровно один из следующих типов: входная, выходная, пустая, целевая или управляющая.

В соответствии с ограничениями работы, все управляющие команды будут являться командами вызовов функций.

Одной из форм сигнатуры потока управления процесса является набор связанных между собой графов потока вызовов, один из которых соответствует функции, с которой начинается исполнение процесса. Под связыванием графов понимается установка соответствия между вершинами вызовов функций и графами потока управления этих функций. В работе предлагается способ мониторинга потока управления процессов в операционных системах, основанный на использовании данной формы сигнатуры потока управления.

3 Граф путей потока управления

Алгоритм сравнения фактического потока управления с эталонной моделью в виде сигнатуры потока вызовов состоит из двух этапов: получение от операционной системы очередной исполненной процессом целевой команды и проверка соответствия этой команды сигнатуре. Таким образом, перед проверкой очередной исполненной целевой команды алгоритм сравнения должен получить множество допустимых сигнатурой целевых команд. Для получения данного множества необходимо провести поиск текущих допустимых целевых инструкций в сигнатуре. Для осуществления поиска предлагается совершать проход по сигнатуре, основанный на обходе графов потока вызовов. В качестве алгоритма обхода графов используется обход в ширину.

При обходе графа потока вызовов могут возникать разветвления, вызванные, например, условными операторами. В силу того, что информация о фактическом потоке управления процесса доступна только в объеме целевых инструкций, проход по сигнатуре должен учитывать все разветвления потока управления. Обход по графу потока вызовов должен завершаться при достижении непустых вершин. Результатом обхода являются ветви потока управления, каждая из которых представляет собой некоторый путь от вершины, с которой был начат обход, до непустой вершины, на которой обход завершился. Дальнейшие шаги алгоритма поиска для каждой полученной ветви потока управления зависят от типов вершин, на которых ветви остановились.

При встрече в ветви потока управления целевой вершины, ее дальнейшую обработку следует остановить. Соответствующий данной вершине целевой вызов попадает в множество допустимых целевых инструкций сигнатуры на данном шаге прохода. При встрече в ветви потока управления управляющей вершины, соответствующей вызову некоторой функции, требуется совершить аналогичный обход по графу этой функции. Такое действие будем называть вызовом графа функции. Наконец, при встрече в ветви потока управления выходной вершины, требуется переместить ветвь потока управления в ту вершину вызвавшего функцию графа, в которой данный вызов был произведен. После этого требуется продолжить обход исходного графа потока вызовов до встречи очередных непустых вершин. Такое действие будем называть возвратом из графа функции.

Таким образом, проход по сигнатуре подразумевает за собой перенос обхода в графы других функций (вызовы функций) и возврат обхода из них в исходную вершину (возврат из функции). При этом при совершении возврата из функции ранее могло быть совершено несколько вызовов других функций. В силу этого для реализации алгоритма требуется структура данных, по которой возможно определять вершину возврата из функции по достижении выходной вершины графа этой функции.

Тривиальным вариантом такой структуры данных может являться стек, соответствующий стеку вызовов функций, совершенных во время прохода по сигнатуре. Основным недостатком данного подхода является необходимость хранения такого стека для каждой ветви потока управления, полученной во время обхода. На практике такой подход оказался неприменим в силу количества формируемых стеков вызовов. Помимо этого, такая структура данных не позволит обрабатывать рекурсивные функции в общем случае. Пусть некоторая функция после входной вершины имеет разветвление в целевую вершину или вершину рекурсивного вызова. Тогда при обходе такого графа необходимо вновь начать обход

этого графа от входной вершины, который так же приведет к данной вершине вызова. Так как количество рекурсивных вызовов, вообще говоря, не ограничено, количество формируемых стеков для данного разветвления также будет неограниченным.

Для решения данных проблем предлагается использовать структуру данных, называющуюся графом путей потока управления. Граф путей представляет собой орграф, каждая вершина которого однозначно соответствует единственной вершине из графов потока вызовов сигнатуры, при этом вершины графа путей связаны между собой в соответствии с разветвлениями потока управления данной сигнатуры. Данный граф позволяет определять, в какие вершины должен возвращаться поток управления при выходе из функций. Для совершения поиска допустимых целевых команд для графа вводятся операции над вершинами, соответствующие обходу функции от данной вершины, вызову функции данной вершины и возврату из функции данной вершины. Проблема прохода рекурсивных функций решается с помощью использования в графе циклов для запоминания вершин возврата из рекурсивных функций. Для исключения повторного вызова уже пройденных функций используются атрибуты вершин графа путей.

4 Заключение

В результате работы предложена структура данных и алгоритмы над ней, которые позволяют на каждом шаге мониторинга потока управления процесса в операционной системе получать набор допустимых целевых инструкций для данного состояния потока управления. В рассмотренном случае процесс является однопоточным, его поток управления не может быть изменен системными прерываниями, а его программа не содержит команды перехода за исключением вызовов функций. Полученные структура данных и алгоритмы могут быть использованы для мониторинга процессов более общего вида, включая многопоточность, системные прерывания и переходы.

Литература

1. Патент № 2817547 RU, МПК G06F 12/14 (2006.01), G06F 21/52 (2013.01). Система и способ контроля работоспособности процессов в операционной системе : № 2023130106 : заявлен 20.11.2023: опубликован 16.04.2024 / Сорокин И.А., Пучкин Д.А., Духвалов А.П. ; заявитель Акционерное общество «Лаборатория Касперского» (RU)