

Уте́чка па́мяти (англ. *memory leak*) — процесс неконтролируемого уменьшения объёма свободной оперативной или виртуальной памяти компьютера, связанный с ошибками в работающих программах, вовремя не освобождающих ненужные участки памяти, или с ошибками системных служб контроля памяти.

Ссылки

```
1  /*1*/ char *pointer = NULL;
2  /*2*/ for( int i = 0; i < 10; i++ ) {
3  /*3*/     pointer = new char[100];
4  /*4*/ }
5  /*5*/ delete [] pointer;
```

Чем опасны утечки памяти

Динамическая память является ограниченным ресурсом. Управление динамической памятью программы обычно осуществляется библиотекой языка программирования, которая сама работает поверх динамической памяти, предоставляемой операционной системой.

Утечки памяти приводят к тому, что потребление памяти программой неконтролируемо возрастает, в результате рано или поздно вступают в действие архитектурные ограничения среды исполнения (операционной системы, виртуальной машины, ЭВМ), и тогда новое выделение памяти становится невозможным. В этой ситуации в программе, которая запрашивает память, обычно происходит аварийный останов. Это может по стечению обстоятельств произойти и совсем с другой программой после того, как программа, подверженная утечкам, исчерпает всю память ЭВМ.

Способы предотвращения

Существуют различные способы предотвращения утечек памяти.

Отказ от динамической памяти

Например, FORTRAN-77 полностью отказывается от применения механизмов динамического распределения памяти, что исключает подобные ошибки, но существенно ограничивает функциональность программ.

Владеющие указатели

Владеющие указатели позволяют в той или иной мере согласовать время жизни указателя и время жизни объекта, на который он ссылается. Тем не менее, использование владеющих указателей не помогает в случае циклических ссылок между объектами. (подробнее см. паттерн «Получение ресурса есть инициализация»)

Сборка мусора

Некоторые языки программирования (например, Оберон, Java, языки платформы .NET) предоставляют средства, позволяющие автоматически освобождать неиспользуемую память («сборщик мусора», англ. *garbage collector*). Сборщики мусора решают также и проблему циклических ссылок, но сборка мусора является ресурсоёмкой операцией. За использование подобных средств приходится расплачиваться быстродействием системы, и, главное, сборка мусора вносит неожиданные паузы в программу, что недопустимо в системах реального времени.

Сборка мусора была изобретена Джоном Маккарти примерно в 1959 году при разработке языка программирования Лисп, структура которого делает крайне затруднительным ручное управление памятью.

Перезапуск программы

В тех случаях, когда устранить утечки памяти не представляется возможным, например, при использовании кода, поставляемого в виде программных модулей и изготовленного сторонними разработчиками, применяют своеобразный способ игнорирования утечек. Код, подверженный утечкам, размещают в отдельной программе, а эту программу с нужной периодичностью перезапускают. Запуски и перезапуски программы выполняются внешней программой, которая

также подаёт исходные данные и забирает результаты. Поскольку при завершении программы вся память, затребованная ей у операционной системы, возвращается операционной системе, такой метод не позволяет утечкам приобрести катастрофический характер.

Утечка других ресурсов

Также существует ошибка, именуемая утечкой дескрипторов: захваченные дескрипторы не возвращаются операционной системе.

Для борьбы с последствиями таких ошибок разработчики операционных систем вводят в них функциональность, позволяющую ограничивать объём памяти, количество дескрипторов и количество процессорного времени, доступного одному пользователю или конкретному процессу.

Обнаружение утечек

Для профессиональных языков программирования существуют специальные программы-профилировщики, позволяющие обнаружить в числе прочего и утечки памяти.


Для некоторых языков программирования существуют статические анализаторы кода, выявляющие элементы программы, потенциально способные приводить к логическим ошибкам, в том числе и к утечке памяти. Примитивный вариант такого анализатора реализует практически любой компилятор языка высокого уровня, в виде выдачи так называемых предупреждений (warnings) — сообщений о наличии в программе конструкций, формально не нарушающих синтаксис языка, но потенциально ошибочных.

Существуют библиотеки для отладки использования памяти, помогающие следить за выделением и освобождением памяти во время работы программы.

См. также

- Динамический анализ кода
- Dmalloc
- Valgrind
- totalview

Ссылки

- Memory Leak Detection in Embedded Systems (<http://linuxjournal.com/article.php?sid=6059>)
Архивировано (<https://archive.today/20121209063515/http://linuxjournal.com/article.php?sid=6059>)
9 декабря 2012 года.
- Java memory leaks — Catch me if you can (http://www.ibm.com/developerworks/rational/library/05/0816_GuptaPalanki/)
- Sample Chapter from Bitter Java (<http://java.sun.com/developer/Books/javaprogramming/bitterjava/bitterjavach06.pdf>) 
- Memory Leaks, Be Gone (http://dev2dev.bea.com/pub/a/2005/06/memory_leaks.html)
- Memory Management in Objective-C (<http://www.macdevcenter.com/pub/a/mac/2001/07/27/cocoa.html>)
- Борьба с утечками ресурсов в реальном времени (<http://www.xakep.ru/post/41631/default.asp>)

Источник — https://ru.wikipedia.org/w/index.php?title=Утечка_памяти&oldid=114026435

Эта страница в последний раз была отредактирована 5 мая 2021 в 17:59.

Текст доступен по лицензии Creative Commons Attribution-ShareAlike; в отдельных случаях могут действовать дополнительные условия.

Wikipedia® — зарегистрированный товарный знак некоммерческой организации Wikimedia Foundation, Inc.