Секция безопасности информационных технологий

УДК 681.3.067

Л.К. Бабенко, О.Б. Макаревич, Д.Э. Лим

ПРОЕКТИРОВАНИЕ БЕЗОПАСНЫХ МНОГОПРОЦЕССОРНЫХ СИСТЕМ

В данной работе рассматривается задача проектирования безопасных многопроцессорных систем.

В соответствии с одной из современных классификаций [1] все системы с MIMD-архитектурой можно классифицировать по структуре памяти и методам синхронизации между процессорами.

В зависимости от структуры оперативной памяти существуют:

1) системы с общей памятью, адресуемой всеми процессорами;
2) системы с распределенной памятью, каждая часть которой доступна только одному процессору.

Межпроцессорные взаимодействия строятся через разделяемые переменные или с помощью механизма передачи сообщений. При этом могут существовать следующие виды архитектур:

1) общая память – разделяемые переменные (GMSV);
2) распределенная память – разделяемые переменные (DMSV);
3) распределенная память – передача сообщений (DMMP);
4) общая память – передача сообщений (GMMP).

Вычислительные системы, использующие общую разделяемую память для межпроцессорного взаимодействия и синхронизации, называются системами с разделяемой памятью, например, CRAY Y-MP [2] (класс 1). Системы с распределенной памятью и синхронизацией через разделяемые переменные, как в BBN Butterfly, называются гибридными архитектурами (класс 2).

Системы, в которых память распределена по процессорам, а для взаимодействия и синхронизации используется механизм передачи сообщений, называются архитектурами с передачей сообщений, например: NCube, EC2703[3] (класс 3). Данный класс МВС является наиболее перспективным в настоящее время.

При проектировании многопроцессорных систем наиболее значимыми с точки зрения обеспечения безопасности являются два направления:

1) поддержание целостности данных при многозадачной и многопроцессорной обработке;
2) защита от несанкционированного доступа к данным.

Другие угрозы для обрабатываемых данных либо значительно менее существенны, либо имеют очень специфические методы решения.

Поддержание логической целостности данных в МВС осуществляется как на уровне ОС, так и на уровне прикладных программ. В самой ОС существует множество структур данных, требующих защиты. Решение данной задачи осуществляет-

ся, в первую очередь, путем корректной разработки системного и прикладного программного обеспечения. Соответствующие средства могут использовать формальные, программнореализованные методы распознавания тупиков. Примером таких средств могут быть:

- семафоры;
- методы, основанные на ведении таблиц распределения ресурсов и таблиц запросов к занятым ресурсам. Анализ этих таблиц позволяет обнаружить взаимные блокировки;
- мониторы (высокоуровневое средство синхронизации).

В МВС-системах, состоящих из нескольких процессоров, каждый из которых имеет собственную оперативную память (класс 3), семафоры и мониторы оказываются непригодными. В таких системах синхронизация может быть реализована только с помощью обмена сообщениями.

Для защиты данных от случайных повреждений используются корректирующие коды.

В распределенных системах схема контроля целостности данных подразумевает выполнение двумя сторонами – *источником* и *приемником* — некоторых (возможно, разных) криптографических преобразований данных. Источник преобразует исходные данные и передает их приемнику вместе с некоторым приложением, обеспечивающим избыточность шифрограммы. Приемник обрабатывает полученное сообщение, отделяет приложение от основного текста и проверяет их взаимное соответствие, осуществляя таким образом контроль целостности. Конкретное криптографическое преобразование является частью общего протокола взаимодействия. В различных средах целостность передаваемых сообщений может обеспечиваться различными сочетаниями основных механизмов – цифровой подписи (например, ГОСТР 34.10–94, ГОСТР 33.11–94 и имитовставки (например, по ГОСТ 28147-87).

В связи с тем, что многопроцессорная система является сосредоточенным объектом, то использовать указанные средства контроля целостности нецелесообразно.

Защита от несанкционированного доступа к данным представляет собой сложную задачу, решение которой может привести к противоречию с основными функциями вычислительной системы. Как известно, удобство пользования вычислительной системой обратно пропорционально степени ее защищенности.

Основной задачей контроля и управления доступом является установление множества операций, разрешённых для пользователей. В настоящее время существует два основных механизма управления доступом — дискреционный (произвольный) и мандатный (нормативный).

Произвольное управление доступом

Произвольное управление доступом в соответствии с требованиями «Оранжевой Книги» появляется начиная с класса С.

Основой произвольного управления доступом является матрица прав доступа, строки которой соответствуют субъектам (пользователи, процессы и т.д.), а столбцы объектам (файлы, каталоги, процессы и т.д.). В ячейках матрицы содержатся права доступа субъектов к объектам.

В качестве субъектов в МВС могут выступать пользователи, между которыми в общем случае динамически распределяются ресурсы решающего поля (РП). В работе не первом этапе рассматривается статическое распределение ресурсов РП.

С другой стороны, процессы, развивающиеся в РП, выступают как субъекты при обращении к параллельной системе баз данных.

В зависимости от способа представления матрицы прав доступа в ОС различают несколько способов реализации произвольного контроля доступа. Наиболее распространёнными для операционных систем являются:

1. списки прав доступа (Access Control List — ACL);
2. биты доступа.

Списки прав доступа

При произвольном управлении доступом с помощью ACL с каждым объектом ассоциируется список пользователей, в котором указаны их права доступа к объекту. При принятии решения о доступе соответствующий объекту доступа ACL проверяется на наличие прав, ассоциированных с идентификатором пользователя, запрашивающего доступ, или процесса, выступающего с запросом к параллельной системе баз данных.

Биты защиты

Вследствие того, что многие защищённые ОС ведут своё происхождение от UNIX, они реализуют произвольный доступ с помощью механизма битов защиты. При этом вместо списка пользователей, которым разрешён доступ к объекту, с объектом связываются биты защиты. В ОС UNIX биты защиты указывают права доступа субъектов по чтению, записи и выполнению. При этом биты защиты может изменять только владелец задачи и администратор.

В МВС, также как и в других современных системах, предполагается использовать комбинации списков контроля доступа и битов защиты.

Нормативное управление доступом

В отличие от произвольного управления доступом, которое позволяет передавать права одного пользователя другому, нормативное управление доступом полностью запрещает передачу прав доступа между пользователями. Это позволяет разрешить проблему «троянских коней» в защищённых информационных системах. Нормативное управление доступом, основанное на модели Белла–Лападула и известное как правила «запрета чтения с верхнего уровня» и «записи на нижний уровень», может быть перенесено в МВС с соответствующими модификациями, учитывающими архитектурные особенности системы.

Если MIMD-МВС относится к классам 3 и 4, т.е. процессы взаимодействуют посредством передачи сообщений (DMMP, GMMP), и отношения между компонентами МВС построены на основе модели клиент/сервер, то задачи по обеспечению безопасности решаются службой безопасности на уровне ОС. Наиболее перспективной с этой точки зрения является технология реализации ОС, получившая известность как микроядерная [2], [3].

Все компоненты системы используют средства микроядра для обмена сообщениями, но взаимодействуют непосредственно. Микроядро лишь проверяет за-

конность сообщений, пересылает их между компонентами и обеспечивает доступ к аппаратуре.

В результате такие важные компоненты ОС, как файловая система, сетевая поддержка и т.д., превращаются в по-настоящему независимые модули, которые функционируют как отдельные процессы и взаимодействуют с ядром и друг с другом на общих основаниях.

Этот подход построения операционных систем использовался и в авторской разработке отечественной МВС ЕС2703 [4].

Структура и организация данной МВС позволяет функционировать в вычислительной среде множеству параллельных процессов различных типов. Организация ОС МВС выполнена по принципу разделения основных функций и реализации их на собственных процессорах. Ядро ОС осуществляет инициализацию среды реального времени, управление службой времени, планирование задач на выполнение в соответствии с масштабом реального времени, загрузку задач и управление их прохождением по ВС, а также обеспечивает взаимодействие с файловой системой для формирования связей задач с файлами данных с учетом координации коммутирующих программ через процессор обмена. Основными преимуществами ОС, реализованной на собственных процессорах, являются:

- повышение ее производительности;
- отсутствие аппарата ключей защиты, так как выполнение системных программ и программ пользователя производится на независимых процессорах с локальной памятью;
- управление режимом мультипроцессной обработки на РП, осуществляемое ядром ОС, аппаратно разделенным с РП, что дает возможность освободить макропроцессоры от системной составляющей, и приведет к довольно простой структуре ОС.

ЛИТЕРАТУРА

1. *Воеводин В.В., Капитонова А.П.* Методы описания и классификации вычислительных систем. М.: МГУ,1994.
2. *Зегжда Д.П., Ивашко А.М.* Как построить защищенную информационную систему. СПб: «Интерлайн», 1998.
3. *Каляев А.В., Николаев И.А., Макаревич О.Б., Бабенко Л.К.* Супер ЭВМ с программируемой архитектурой. // ЭВТ. Сб. Статей. М.: Радио и связь, 1988. Вып.2.
4. *Бабенко Л.К., Макаревич О.Б., Матвеева Л.Н.* Принципы организации вычислений в многопроцессорных системах для приложений реального времени. Препринт № 16-88. Львов, 1988.

УДК 681.325.5

П.А. Федоров, А.К. Шилов, Д.А. Шилов

АППАРАТНАЯ РЕАЛИЗАЦИЯ АЛГОРИТМА СЖАТИЯ ИЗОБРАЖЕНИЙ НА ПРОЦЕССОРЕ TMS320C30

Рассмотрен алгоритм адаптивного полиномиального сжатия изображений с потерями. Выполнен анализ его реализации различными способами.