Классификация механизмов IPC и их проектирование для микроядра общего назначения Sol

Басков Евгений Сергеевич
ИСП РАН (109004, г. Москва, ул. А. Солженицына, дом 25.)
baskov@ispras.ru

Данный доклад посвящен механизмам межпроцессного взаимодействия (Interprocess Communication, IPC) в ядрах операционных систем (ОС). Межпроцессное взаимодействие используется для обмена данными между процессами внутри системы. Оно играет ключевую роль в микроядерных системах, поскольку большая часть системных сервисов переносится из ядра в пользовательские сервисы для увеличения безопасности и устойчивости ОС. В таких ОС от производительности и безопасности IPC зависит безопасность и производительность всей системы в целом, из-за чего интерфейсы таких ОС, как правило, являются более продвинутыми. Поэтому механизмы IPC микроядерных ОС рассматриваются более подробно в данном докладе.

Доклад рассматривает достаточно полную классификацию существующих механизмов IPC по следующим критериям: адресация получателя, способы передачи данных и потока управления, направление передачи, тип передаваемых данных (байты, сообщения, права), блокирование, буферизация. Особое внимание уделяется аспектам безопасности и производительности: ограничениям использования ресурсов (памяти ядра, сервера и процессорного времени), защите от возможных атак и ошибок, идентификации объектов сервера, клиентов и соединений, а также учёту асимметричного доверия — проверке аргументов и ожидания субъекта с меньшими правами. Рассмотрены варианты реализации и проблемы ожидания с ограничением по времени, а также возможность безопасного использования прав как аргументов и идентификаторов объектов сервера.

Далее формулируются сценарии применения механизмов IPC в целом и особенности сценариев использования в микроядерных системах: потоки данных, RPC, компартментализация, доставка событий и уведомлений. Определяются наиболее подходящие формы механизмов IPC для каждого сценария.

Подходы к реализации IPC в различных системах имеют особенности и различные механизмы, подходящие для разных задач. Полная классификация подходов позволяет упростить проектирование новых систем. Для таких систем требуется спроектировать минимальный набор механизмов, которые для заданного набора сценариев использования будут:

- **Полнота**. Предоставлять все необходимые функции, в том числе позволять реализацию и/или эмуляцию требуемых интерфейсов, например, POSIX.
- **Минимальность**. Иметь минимально перекрывающиеся сценарии использования и быть максимально простыми с точки зрения набора возможностей, использования и реализации.
- **Производительность**. Иметь наибольшую эффективность для своих сценариев.
- **Безопасность**. Позволять реализовывать требуемые взаимодействия с учетом требований safety и security.

Также данная классификация позволяет сравнивать операционные системы с точки зрения реализации механизмов межпроцессного взаимодействия. Многие механизмы в ОС не описаны явно и достаточно полно, что требует анализа исходного кода данных ОС для сравнения. При наличии классификации возможно также более эффективно предоставлять классификацию IPC-интерфейсов для новых систем.

Приводится анализ существующих механизмов IPC в рамках рассмотренной классификации. Классификация включает как механизмы монолитных систем на примере **POSIX** IPC, **Linux io_uring** и **Android Binder**, так и микроядерных систем общего назначения и реального времени: **Mach**, **QNX**, **L4**, **seL4**, **EROS**, **Fuchsia Zircon**, **Fiasco.OC**, **Genode**, **Managarm**, **Composite**, **NOVA**, **CLOS** и **ARINC 653** IPC. **Mach** IPC также используется в **macOS** и **iOS**.

В последней части доклада рассматривается реализация IPC в высокопроизводительном экспериментальном микроядре общего назначения **Sol**, разрабатываемом в ИСП РАН, и включается в общую классификацию. Рассматриваются особенности, которые являются улучшениями по сравнению с другими микроядерными системами, в том числе групповые системные вызовы, удаление переключений контекста, безопасное переключение контекстов, эффективный механизм асинхронных уведомлений и механизм ускорения для реализации интерфейсов POSIX для работы с файловой системой. Данная реализация соблюдает баланс между простотой реализации, удобством использования, производительностью, безопасностью и гибкостью, реализуя синхронный интерфейс для удаленного вызова процедур, асинхронный интерфейс для уведомлений и событий, а также механизмы для ускорения передачи больших данных. В частности, важен учёт особенностей архитектуры современных микропроцессоров, возможность эффективной и безопасной реализации интерфейсов **POSIX** и глубокая интеграция *capability-based security* с современными расширениями и дополнительными механизмами безопасности.