Новгородский Государственный Университет им. Ярослава Мудрого

Модули для инструментирования исполняемого кода в симуляторе QEMU

И.А. Васильев

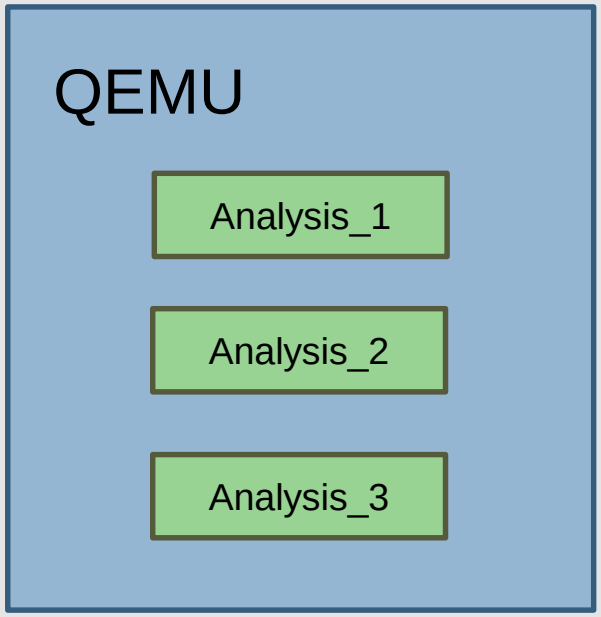
Н.И. Фурсова

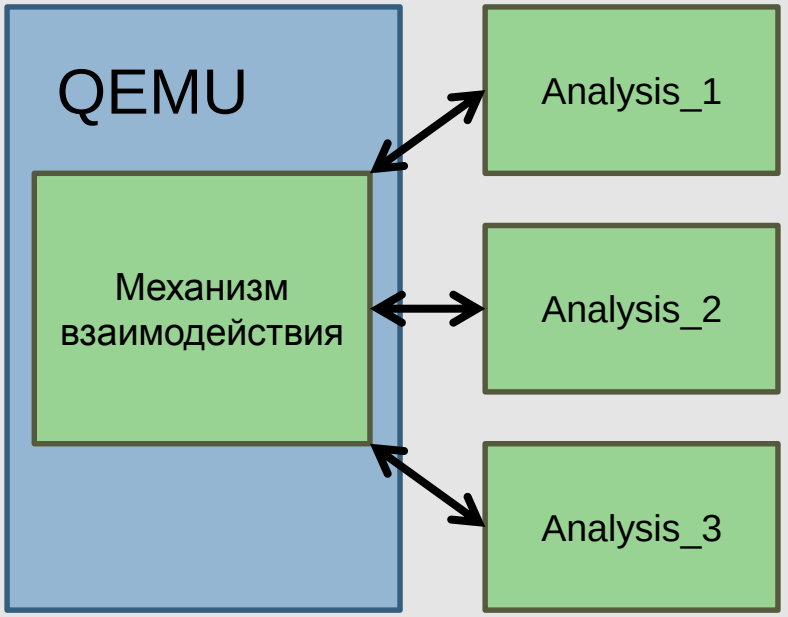
М.А. Климушенкова

П.М. Довгалюк

В.А. Макаров

Полносистемный анализ





- Плохая расширяемость
- Плохая сопровождаемость

- Проблема решена!

Инструментирование

- Инструментирование — техника, которая добавляет в программу код для получения информации о работе программы

- Наш выбор — динамическое бинарное инструментирование

Инструментирование системы

1. DECAF
2. TCG Plugins

Плюсы: богатое API, написание инструментов на языке высокого уровня, инструменты в виде отдельной библиотеки, инструментирование всей системы

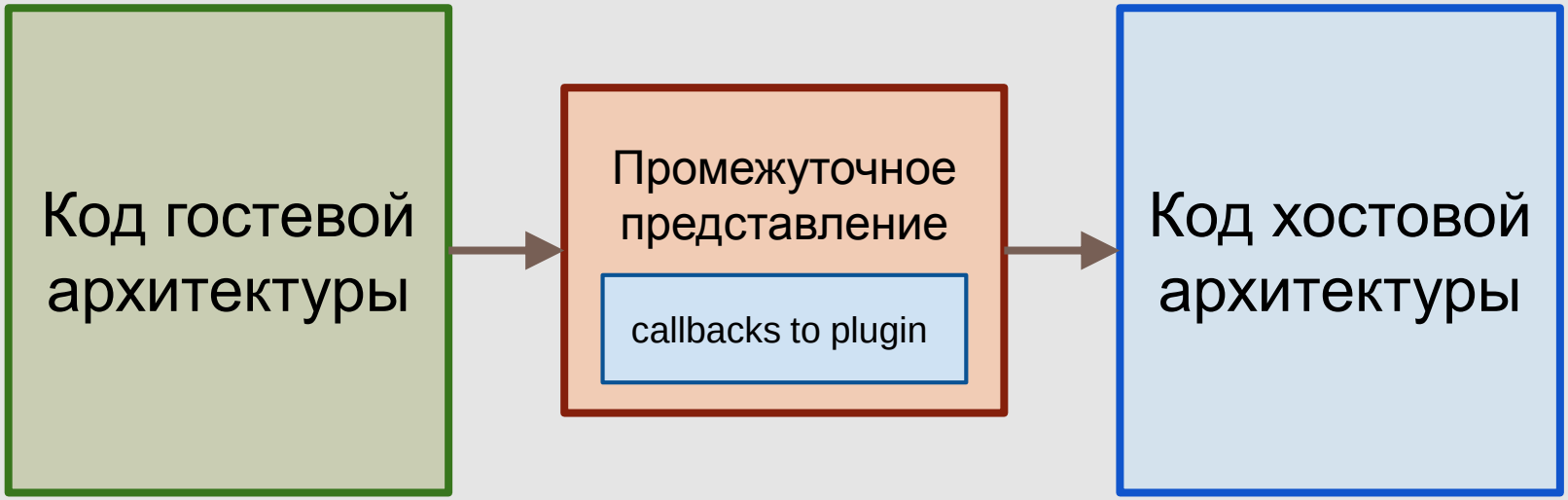
Минусы: значительное замедление скорости работы,неактуальная версия QEMU, интроспекция в ядре QEMU (DECAF)

Требования к реализации

- Необходимость в инструментировании всей системы
- Запуск и остановка инструментирования по команде
- Плагины написанные на языке C
- Возможность подключения сразу нескольких плагинов
- Работа с гостевыми системами как Windows, так и Linux
- Актуальная версия QEMU

Реализация

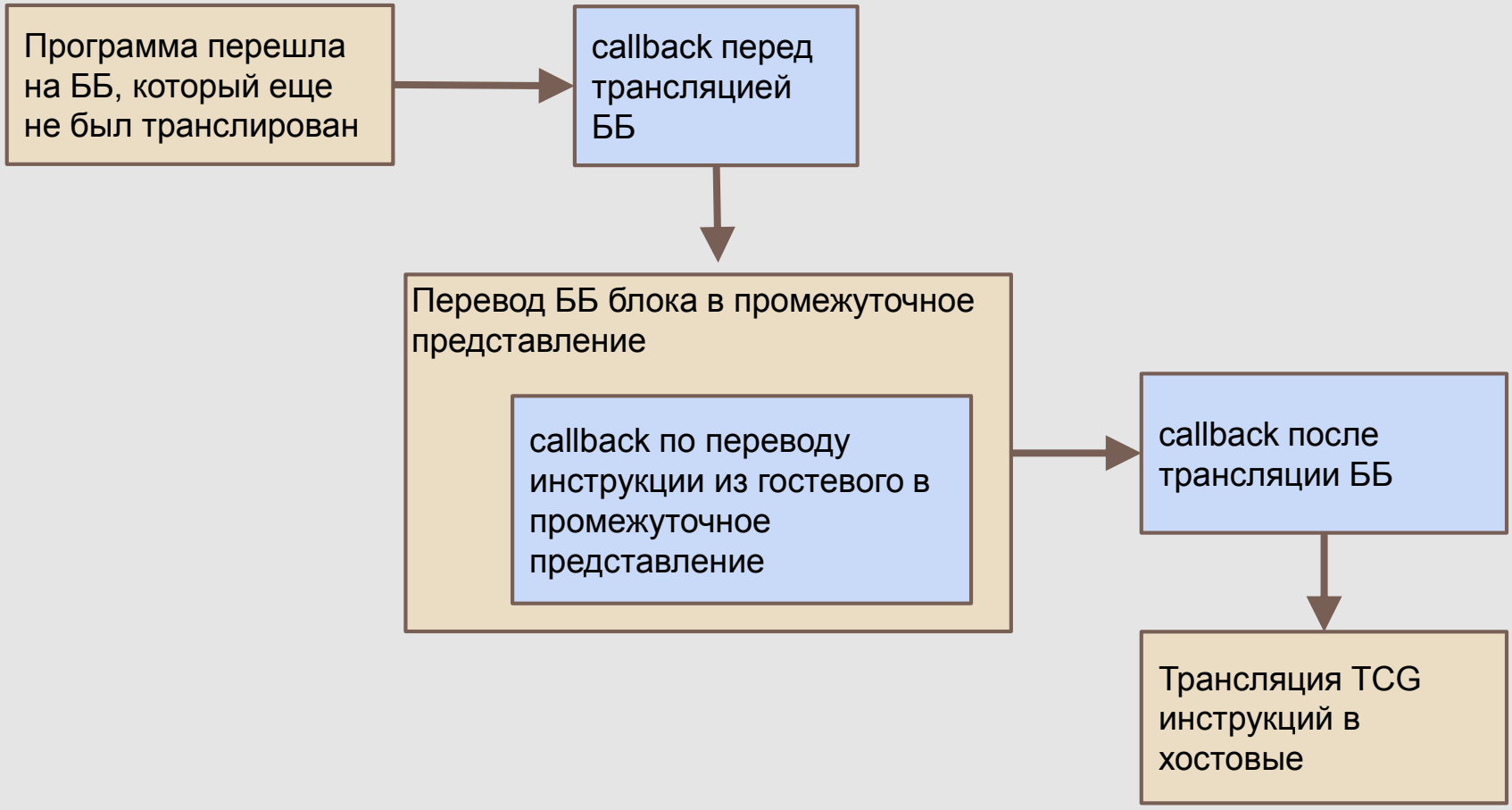
TCG Plugins как основа



Callbacks

- Начало перевода ББ
- Окончание перевода ББ
- Перевод инструкции из гостевого представления в промежуточное
- Окончание работы системы\остановка плагина
- События периферийных устройств
- ...

Инструментирование во время трансляции



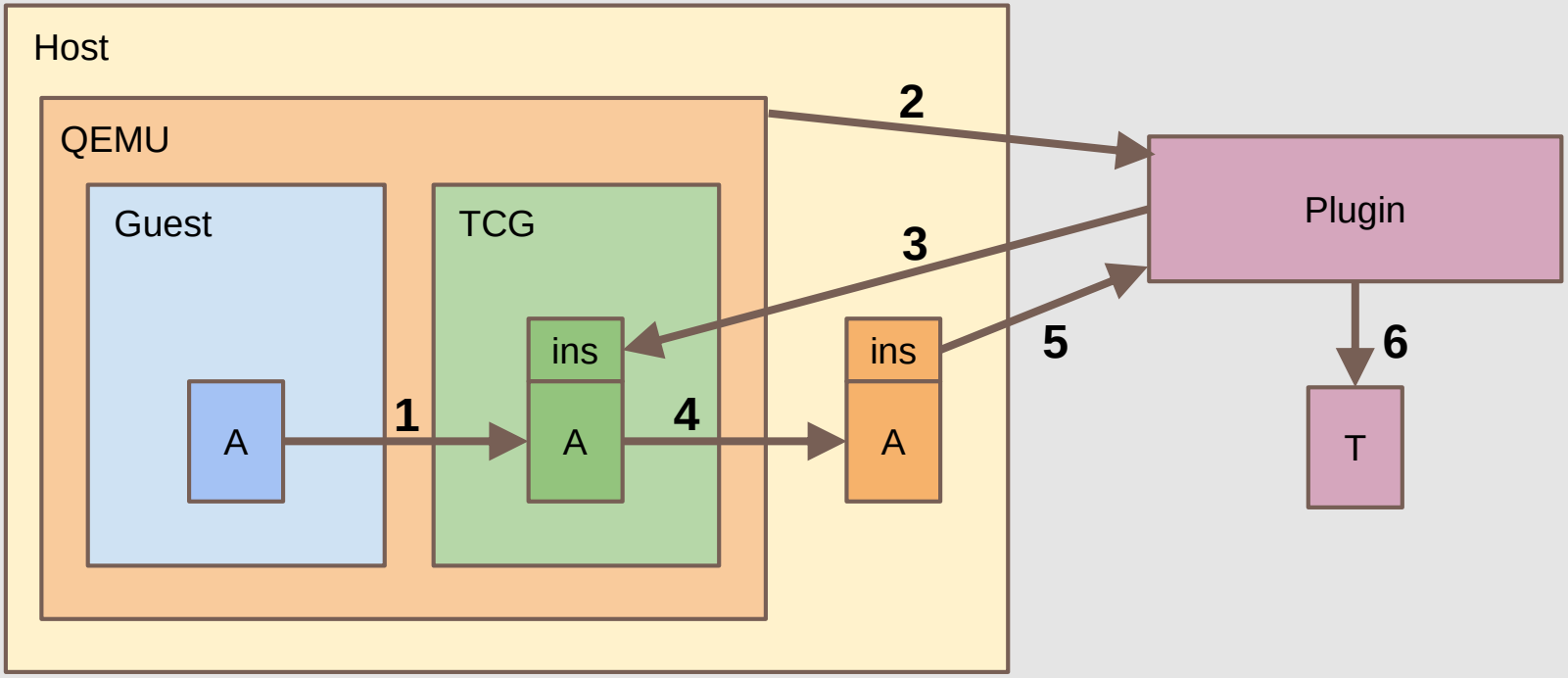
Инструментирование во время выполнения

Реализуется с использованием механизма helper’ов

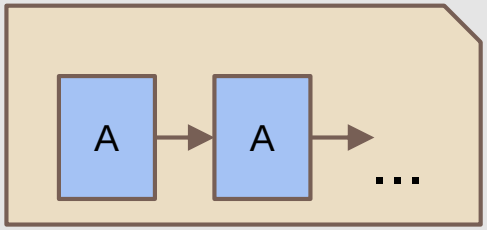
```
//начало базового блока
//гостевая инструкция: orl %ebx, %eax
mov_i32 tmp11, ebx
mov_i32 tmp12, eax
or_i32 tmp13, tmp12, tmp11
//гостевая инструкция:addl $0x01, %eax
movi_i32 tmp14, $0x01
add_i32 tmp15, tmp14, tmp13
mov_i32 eax, tmp15
//окончание базового блока
goto_tb $0x0
```

```
//начало базового блока
//вставить вызов pre_tb_helper
movi_i32 tmp21, $pre_tb_helper_callback
call tmp21, $0x0, $0, env
//гостевая инструкция: orl %ebx, %eax
mov_i32 tmp11, ebx
mov_i32 tmp12, eax
or_i32 tmp13, tmp12, tmp11
//гостевая инструкция:addl $0x01, %eax
movi_i32 tmp14, $0x01
add_i32 tmp15, tmp14, tmp13
mov_i32 eax, tmp15
//окончание базового блока
//вставить вызов post_tb_helper
movi_i32 tmp22,$post_tb_helper_callback
call tmp22, $0x0, $0, env
goto_tb $0x0
```

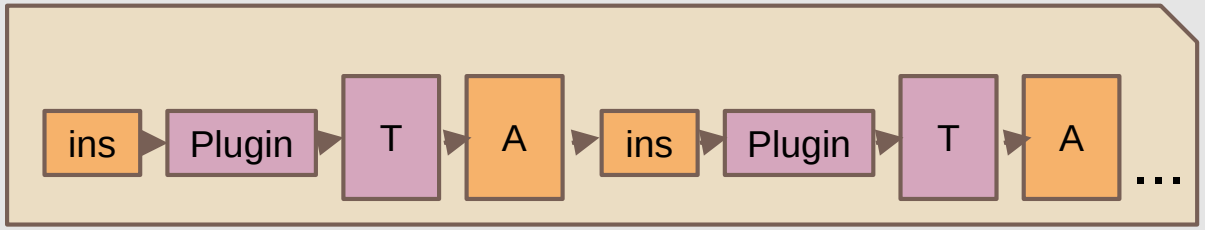
Инструментирование во время выполнения



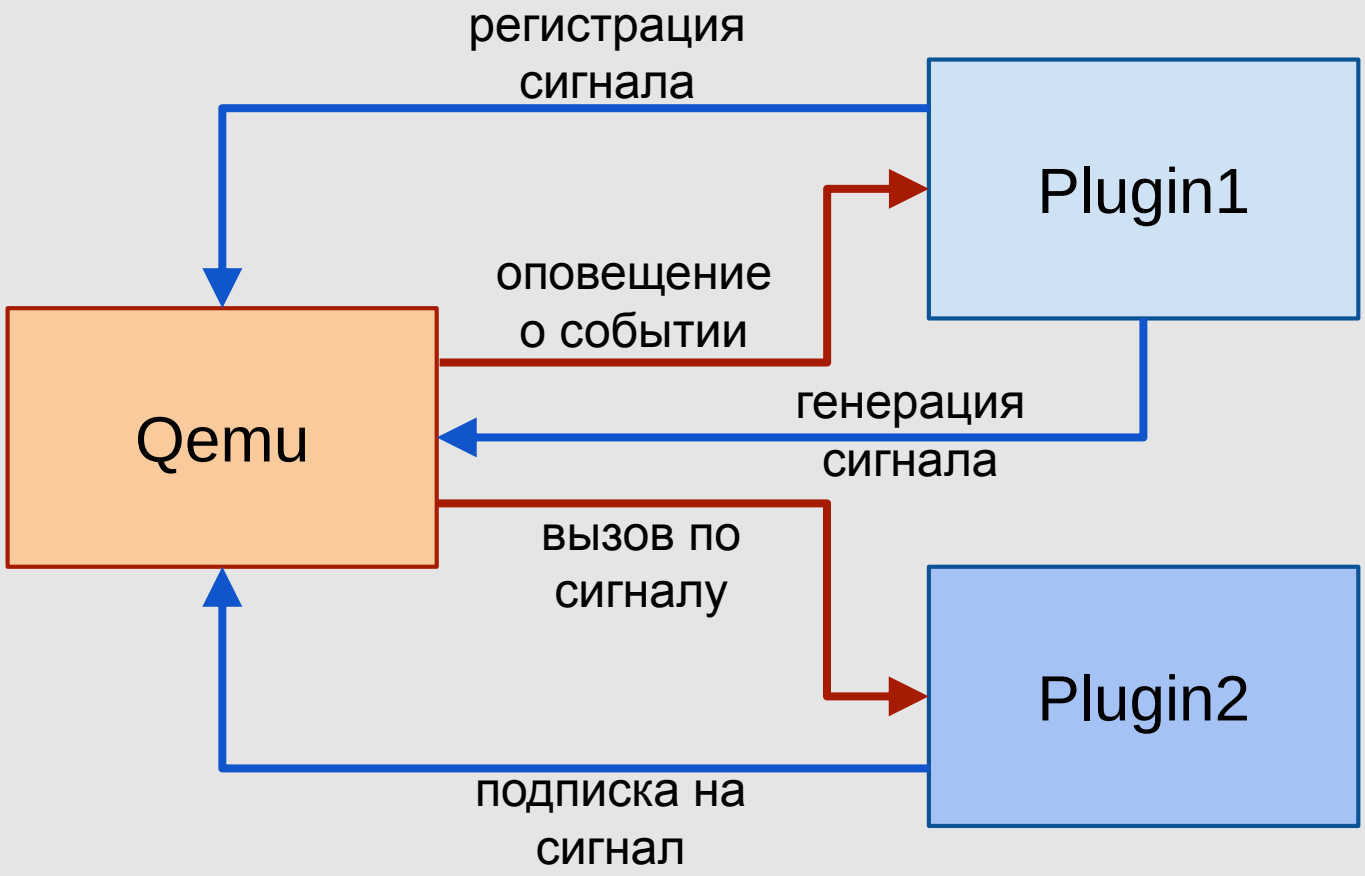
Исходный поток выполнения



Инструментированный поток выполнения



Взаимодействие плагинов



Перспективы

Создание плагинов интроспекции

Примеры плагинов:

-анализ производительности
-отладка памяти
-анализ работы процессов
-анализ работы с файлами
-анализ работы с сетью
-...

Результаты

- Механизм работы плагинов
- Плагин трассировки системных вызовов
- Плагин трассировки гостевых инструкций

Дальнейшие планы

- Расширение API
- Создание плагинов

Спасибо за внимание