# Краткий обзор репозиториев P4 на Github

Здесь кратко описаны репозитории общего пользования p4lang.

Если нужна компиляция исходного кода $P4_{14}$ или $P4_{16}$ для модели поведения bmv2 с использованием консольного интерфейса или thrift API для работы с таблицами, потребуется локально установить и собрать два приведенных ниже репозитория.

- p4c - прототип компилятора $P4_{16}$ (поддерживается также код $P4_{14}$);
- behavioral-model - переопределение модели поведения на языке C++ без автоматического создания кода.

## *Репозитории p4lang*

В Github p4lang является именем «организации», поддерживающей набор репозиториев, связанных с языком P4. Ниже перечислены эти репозитории по состоянию на сентябрь 2020 г.

***behavioral-model***

Переопределение модели поведения behavioral-model (ее часто называют bmv2 - сокращение от Behavioral Model Version 2) на языке C++ без автоматического создания кода.
Первой версией модели служил вывод кода из репозитория p4c-behavioral (компилятор P4 в C). Изменение программы P4 требовало повторной компиляции в новую программу C и последующей ее компиляции. Модель bmv2 больше напоминает интерпретатор для всех возможных программ P4, который настраивает свое поведение в соответствии с конфигурационным файлом bmv2 JSON, создаваемым компилятором p4c. При изменении программы P4 требуется повторная компиляция лишь этой программы без повтора компиляции bmv2.

***education***

P4 для обучения (материалы рабочей группы P4 Education).

***governance***

Wiki с документами проекта P4. В настоящее время репозиторий пуст.

***grpc* (*ветвь grpc/grpc*)**

Обобщенная модель работы с вызовами удаленных процедур (C, C++, Python, Ruby, Objective-C, PHP, C#).

***hackathons***

Код, разработанный на мероприятиях P4 Hackathon.

***mininet* (*ветвь mininet/mininet*)**

Эмулятор для работы с программно-определяемыми сетями (SDN), http://mininet.org.

***ntf***

Платформа для сетевых тестов.

***p4-applications***

Репозиторий рабочей группы P4 Applications, содержащий приложения и документацию по приложениям телеметрии в P4.

***p4-build***

Инфраструктура, требуемая для генерации, сборки и установки библиотеки PD для программ P4.

***p4-hlir***

Написанный на языке Python компилятор для программ $P4_{14}$ (v1.0.x и v1.1.x), генерирующий высокоуровневое промежуточное представление (High Level Intermediate Representation), на основе которого можно создавать компилятор back-end. Создает в памяти объекты Python, представляющие объекты код P4 (заголовки, таблицы, списки полей и т. п.). Выполняет независимые от целевой платформы семантические проверки, включая ссылки на неопределенные таблицы или поля, исключение неиспользуемых таблиц и действий.
p4-hlir не понимает программ $P4_{16}$.

***p4-spec***

Спецификации языка P4 разных версий, PSA (Portable Switch Architecture), INT (Inband Network Telemetry).

***p4app***

Инструменты для сборки, запуска, отладки и тестирования программ P4.

***p4c***

Прототип компилятора (front-end) $P4_{16}$, поддерживающий также программы $P4_{14}$. Репозиторий включает также bmv2 и другие варианты компиляторов back-end.

***p4c-behavioral***

Устаревший компилятор P4 для модели поведения (behavioral model). Заменен behavioral-model.
Компилятор p4c-behavioral использует p4-hlir в качестве front-end для синтаксического анализа исходного кода и дает на выходе промежуточное представление IR, из которого создается код C/C++ для модели поведения версии 1 (не подходит для bmv2).
При установке этого репозитория могут возникать конфликты с p4c-bm, поскольку оба устанавливают модуль Python p4c-bm.

***p4c-bm***

Устаревшая программа генерации конфигурационных файлов JSON для bmv2, а также кода C/C++ PD. Не развивается. Компилятор p4c-bm использует p4-hlir в качестве front-end для синтаксического анализа исходного кода и создает промежуточное представление IR, из которого может быть сгенерирован конфигурационный файл bmv2 JSON. Может также генерировать код C++ PD.

***p4-constraints***

Расширение языка P4 для анонсирования ограничений. См. презентацию. Ограничения могут быть реализованы с помощью библиотеки p4-constraints, представленной в репозитории.

***p4factory***

Устаревшая тестовая реализация спецификации INT (Inband Network Telemetry).

***p4lang.github.io***

Некая информация с сайта P4.org, в том числе анонсы мероприятий.

***p4ofagent***

Агент Openflow для плоскости данных P4.

***p4runtime***
Спецификации P4Runtime - API плоскости управления.

***p4runtime-shell***
Интерактивная оболочка (shell) Python для P4Runtime.

***papers***
Несколько старых статей, связанных с P4.

***PI***
Реализация сервера P4Runtime.

***protobuf*** ***(ветвь protocolbuffers/protobuf)***
Реализация механизмов Protocol Buffers для сериализации структурированных данных от компании Google.

***ptf***
Модель тестирования плоскости данных на основе Python.

***rules_protobuf*** ***(ветвь pubref/rules_protobuf)***
Правила Bazel для создания буферов протоколов и служб gRPC (java, c++, go и т. п.)

***SAI*** ***(ветвь opencomputeproject/SAI)***
Реализация абстрактного интерфейса SAI, определяющего API для независимого от производителя управления элементами пересылки (ASIC, NPU, программные коммутаторы).

***scapy-vxlan***
Реализация scapy от компании Barefoot с поддержкой дополнительных заголовков пакетов, включая VXLAN.

***switch***
Программа switch - пример реализации коммутатора на языке P4, включающая API, SAI и Netlink.

***third-party***
Сторонние программы, от которых зависит p4lang.

***thrift*** ***(ветвь apache/thrift)***
Зеркало репозитория Apache Thrift.

***tutorials***
Учебные материалы по языку P4.

**Николай Малых**
nmalykh@protocols.ru