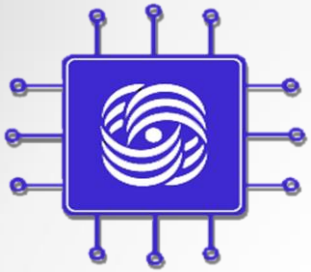


APPLIED
RESEARCH
CENTER FOR
COMPUTER
NETWORKS

Два альтернативных подхода к управлению SDN коммутаторами: OFDPA vs NetConf/YANG

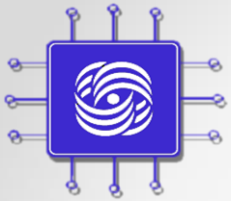
Математический спецкурс
“Программно Конфигурируемые Сети”

к.ф.-м.н., м.н.с., Шалимов А.В.

ashalimov@lvk.cs.msu.su

@alex_shali

@arccnnews



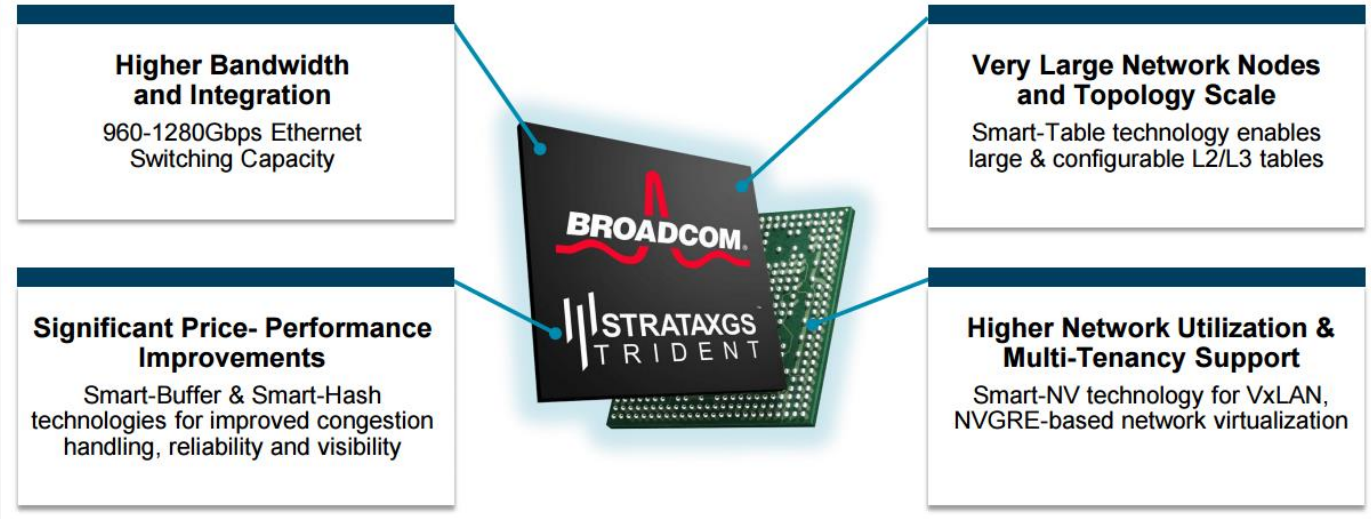
Какие бывают OpenFlow коммутаторы?

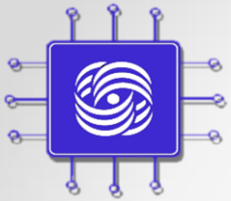
Type	Vendor	Model	Chip
AR	Centec	v350	Centec CTC5163
DR	Foxcon/Caswell	CAT-8020	Tilera TILE-Gx8072
DR	Foxcon/Caswell	CNW3840	Trident2 BCM56854
DR	Corsa	DP6410	Xilinx Virtex FPGAs
DR	Corsa	DP6420	Xilinx Virtex FPGAs
AR	NoviFlow	NoviSwitch 1248	EZchip NP-4 NPUs
DR	NoviFlow	NoviSwitch 2128	EZchip NP-5 NPU
AR	Accton (Edge Core)	AS4610	Broadcom Helix-4
DR	Accton (Edge Core)	AS5710-54X	Broadcom Trident II
DR	Huawei	S12000	Broadcom Trident II
AR	HP	5900	Broadcom Trident+
AR	NEC	PF5240	Broadcom Trident+
AR	Eltex	MF2400	Broadcom Trident
AR	ЦПИКС	Царь	x86
AR	N4B	SWOS	x86
AR	Qtech	Q3800	Broadcom Trident+
DR	Compass Networks	R10004	Ezchip

- FGPA
- NPU
- Hybrid на традиционных chip от Broadcom и Marvell
- X86 с программной начинкой

Какой самый популярный производитель сетевых чипсетов?

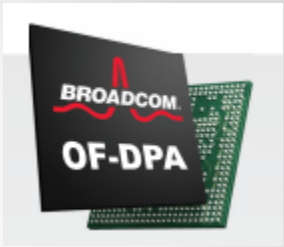
- Конечно, **Broadcom**!
- Обрабатывает 99.98% всего интернет трафика.
- Практически все вендоры (NEC, Extreme, HP, Cisco, Juniper) используют чипсеты от Broadcom.
- ЦОД, корпоративный сегмент, провайдеры



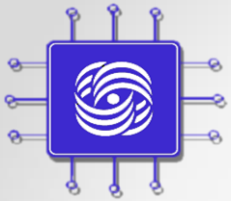


APPLIED RESEARCH CENTER FOR COMPUTER NETWORKS

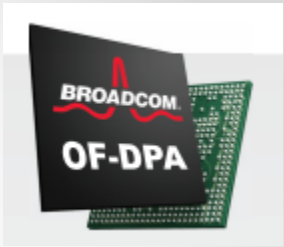
OF-DPA 2.0



- Broadcom OpenFlow Data Plane Abstraction
- Набор программных компонент для чипсетов по трансляции правил OpenFlow в абстракции и конвейер Broadcom чипсетов.
  - SDK, OF-DPA Driver, OF-DPA OpenFlow Agent
- OpenFlow 1.3.4
- Полная поддержка всех features от чипсета: L2/L3 bridging, VXLAN, NVGRE, ACL (drop, rewrite), QoS, queues и т.д.
- Что описывает?
  - Какие таблицы есть, какие поля есть, какие действия, в каких таблицах возможны
  - Experimental значения, чтобы разрешить специфичекую функциональность
- Что получаем?
  - Возможность программировать существующие устройства (после перепрошивки, да)
  - Высокая скорость системы



OF-DPA 2.0



- Broadcom OpenFlow Data Plane Abstraction
- Набор программных компонент для чипсетов по трансляции правил OpenFlow в абстракции и конвейер Broadcom чипсетов.
  - SDK, OF-DPA Driver, OF-DPA OpenFlow Agent
- OpenFlow 1.3.4
- Полная поддержка всех features от чипсета: L2/L3 bridging, VXLAN, NVGRE, ACL (drop, rewrite), QoS, queues и т.д.
- Что описывает?
  - Какие таблицы есть, какие поля есть, какие действия, в каких таблицах возможны
  - Experimental значения, чтобы разрешить специфичекую функциональность
- Что получаем?
  - Возможность программировать существующие устройства (после перепрошивки, да)
  - Высокая скорость системы

Архитектура реализации OF-DPA 2.0

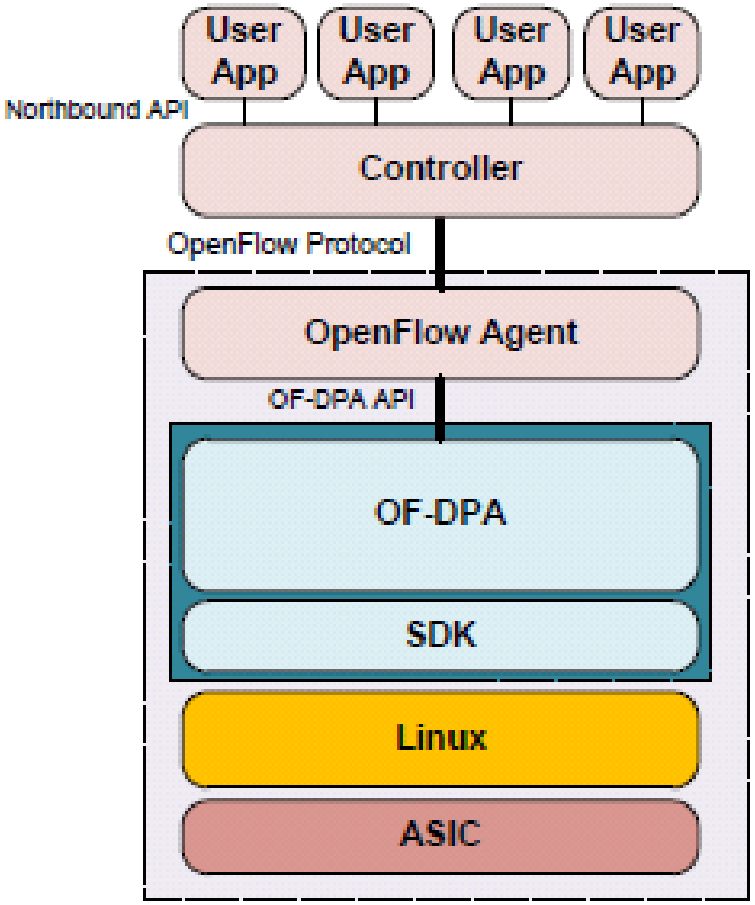
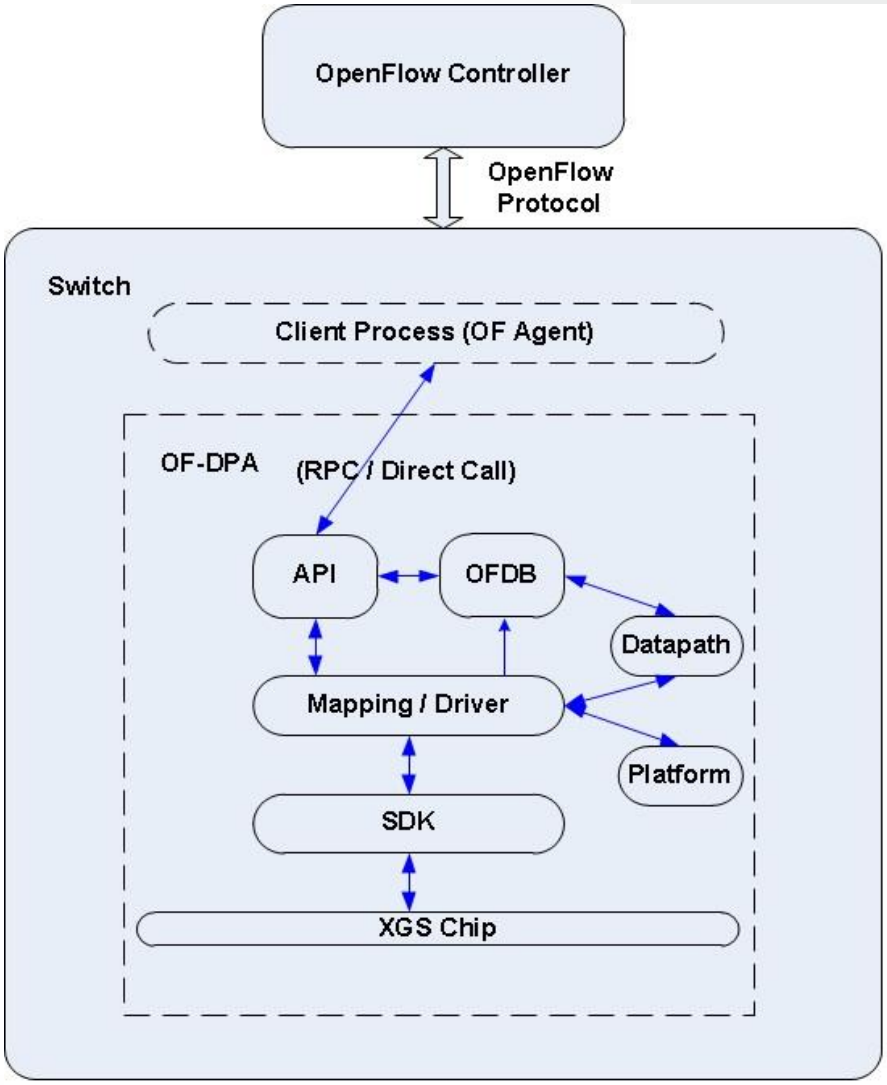
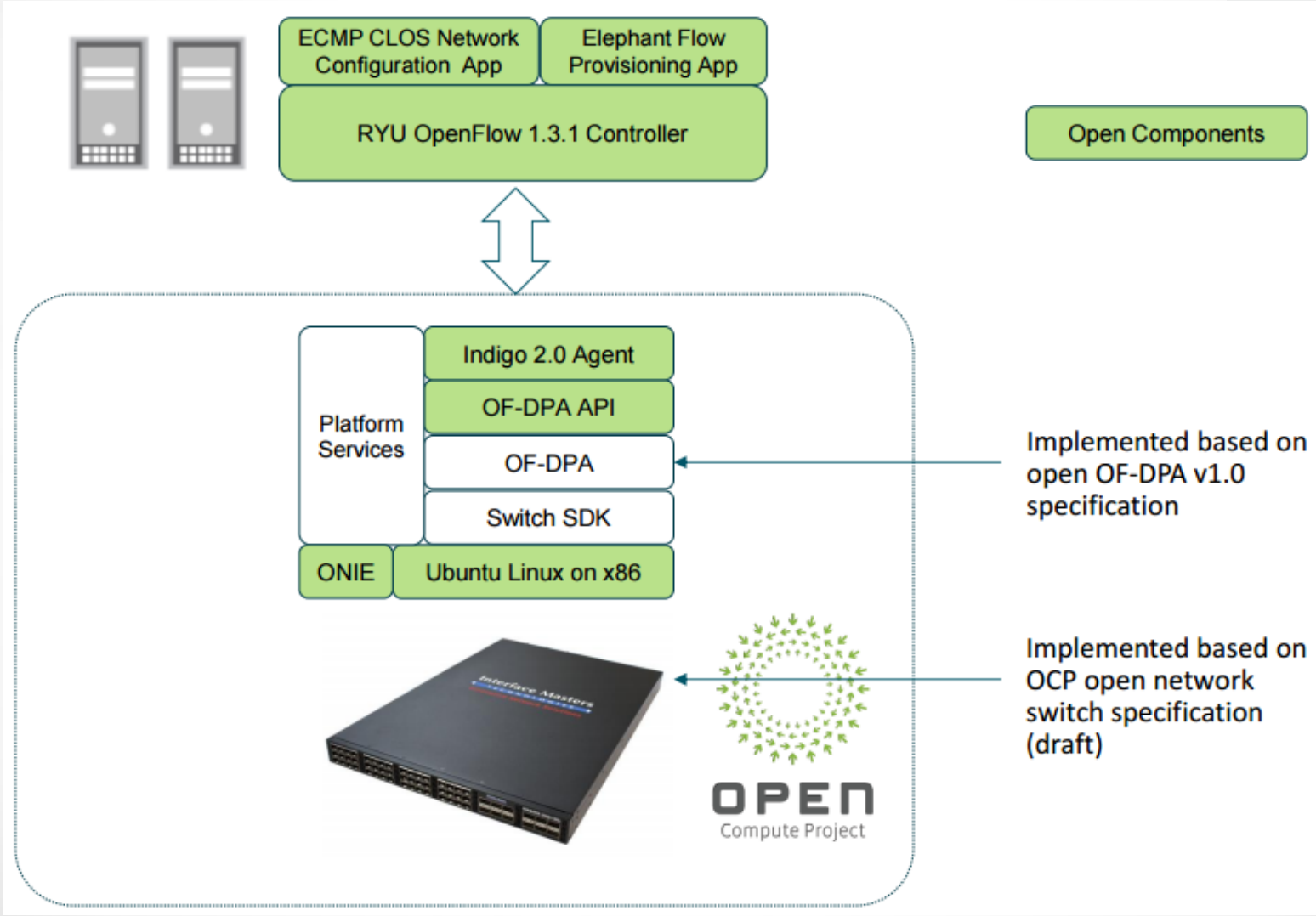
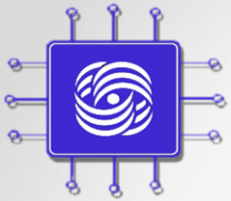


Figure 1: OF-DPA Component Layering

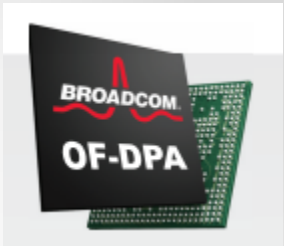


Пример открытого программного стека OF-DPA 2.0



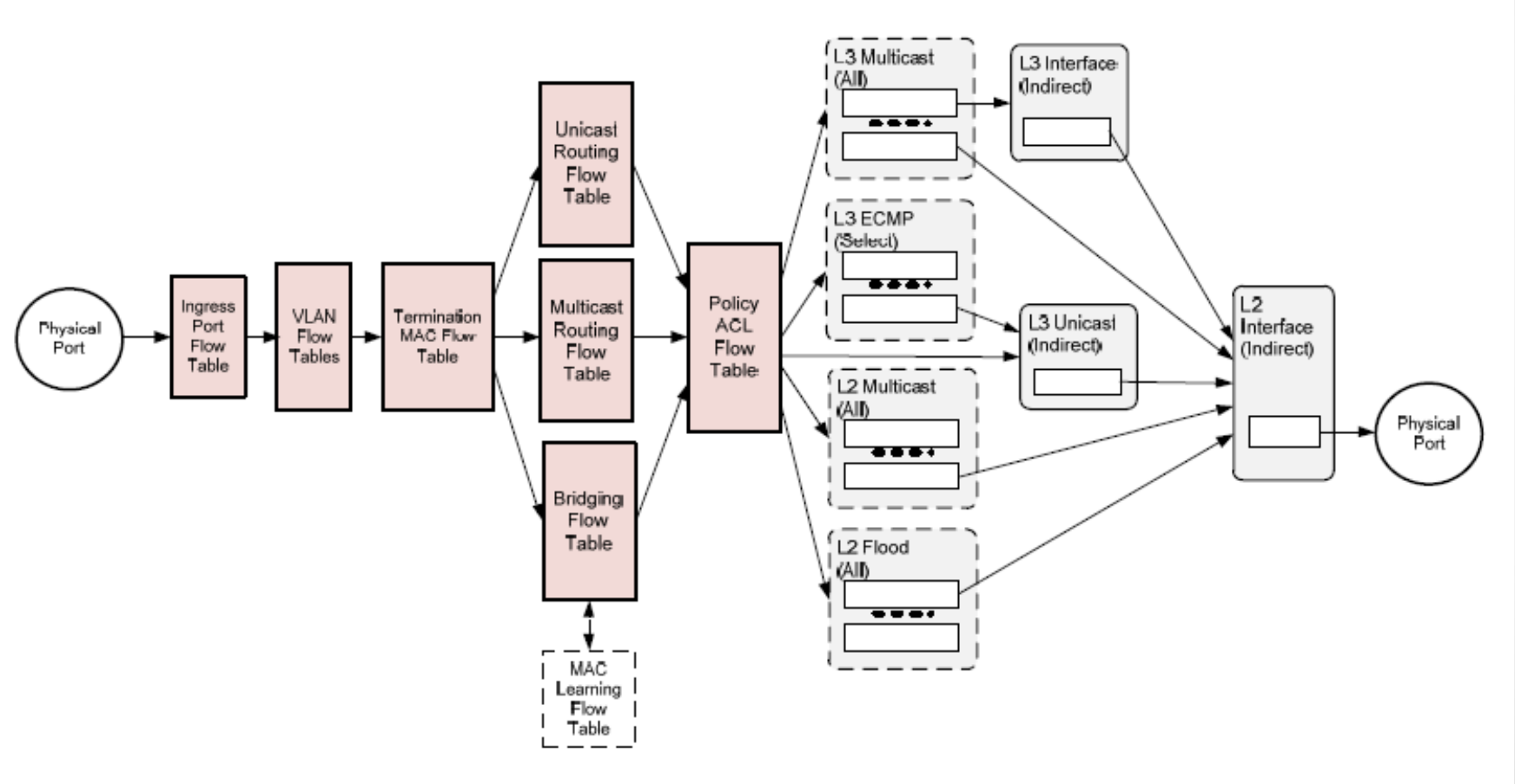


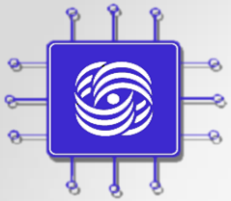
Несколько деталей OF-DPA 2.0



- У таблиц и правил появляется семантика
- Есть несколько типов правил:
  - Стандартные - мы
  - Built-in - зашиты
  - Automatic – добавляются автоматически
- Ограничения на групповые таблицы
  - L3 ECMP, как SELECT
  - L2/L3 broadcast, как ALL
- Можно конфигурировать Switch и на самом OF-DPA SDK. Т.е. самому специфицировать Pipeline без использования SDN/OpenFlow.

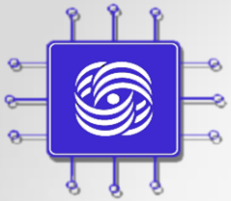
Bridging and Routing OF-DPA 2.0





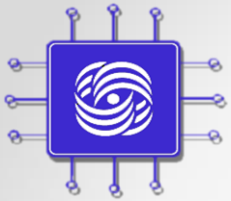
Тра

- Курсы по Операционным системам учат фундаментальным принципам:
  - Примитивы синхронизации, потоки, исключения, файловая система и т.д.
  - Новые языки программирования, операционные системы
- Курса по Сетевым технология учат куче протоколов
  - TCP, UDP, ARP, MPLS, GRE, BGP, OSPF, IS-IS, LDP, RSVP, PIM, ….
  - Отсутствие фундаментальных принципов, только руководства по эксплуатации сетей
  - Алгоритмы маршрутизации одни и теже много лет, управление сетью примитивно




Абстракция

- Курсы по Операционным системам учат фундаментальным принципам:
  - Примитивы синхронизации, потоки, исключения, файловая система и т.д.
  - Новые языки программирования, операционные системы
- Курса по Сетевым технология учат куче протоколов
  - TCP, UDP, ARP, MPLS, GRE, BGP, OSPF, IS-IS, LDP, RSVP, PIM, ….
  - Отсутствие фундаментальных принципов, только руководства по эксплуатации сетей
  - Алгоритмы маршрутизации одни и теже много лет, управление сетью примитивно



APPLIED
RESEARCH
CENTER FOR
COMPUTER
NETWORKS

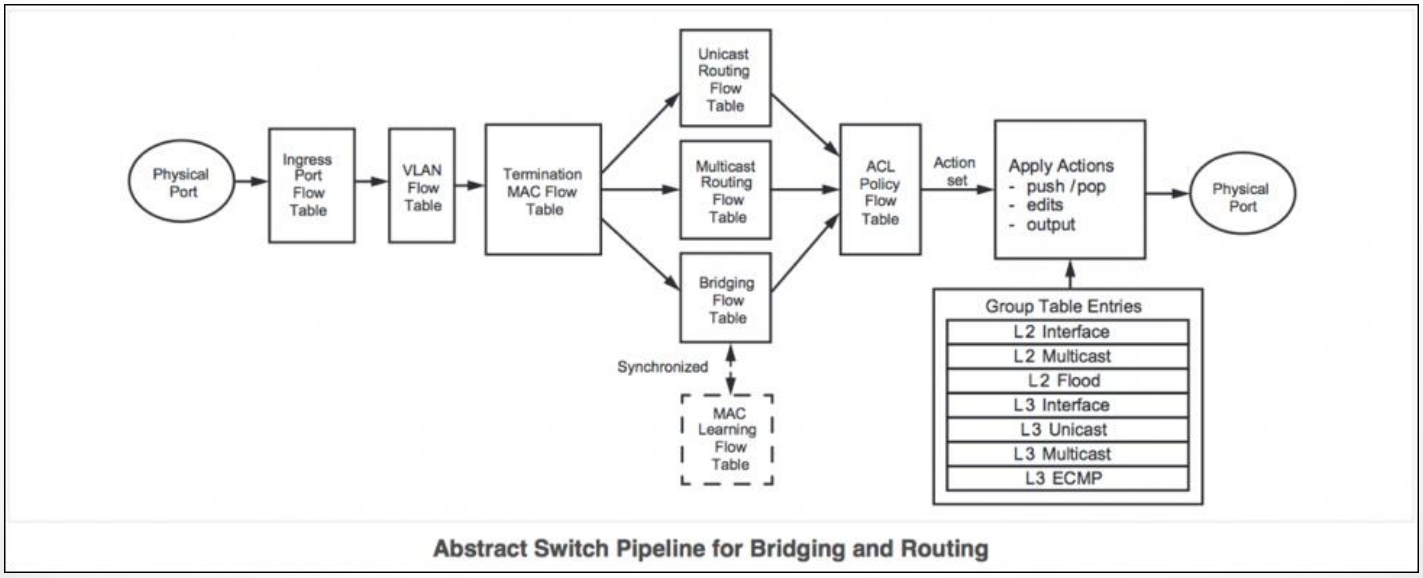
OF-TTP 1.0

(15/08/2014)

Table Type Pattern

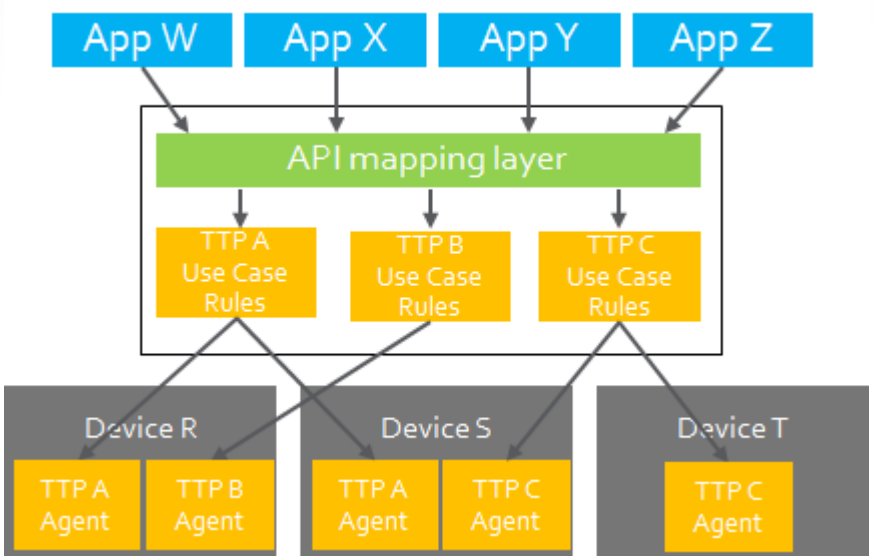
TTP Introduction

- Lifecycle of development, deployment and operation of OpenFlow networks
- To better accommodate heterogeneity of existing hardware switches
- Enable future innovation in hardware switches
- Enable precise communication between app/controller developers and switch vendors
- Enable automated communication between app/controllers and switches



Abstract Switch Pipeline for Bridging and Routing

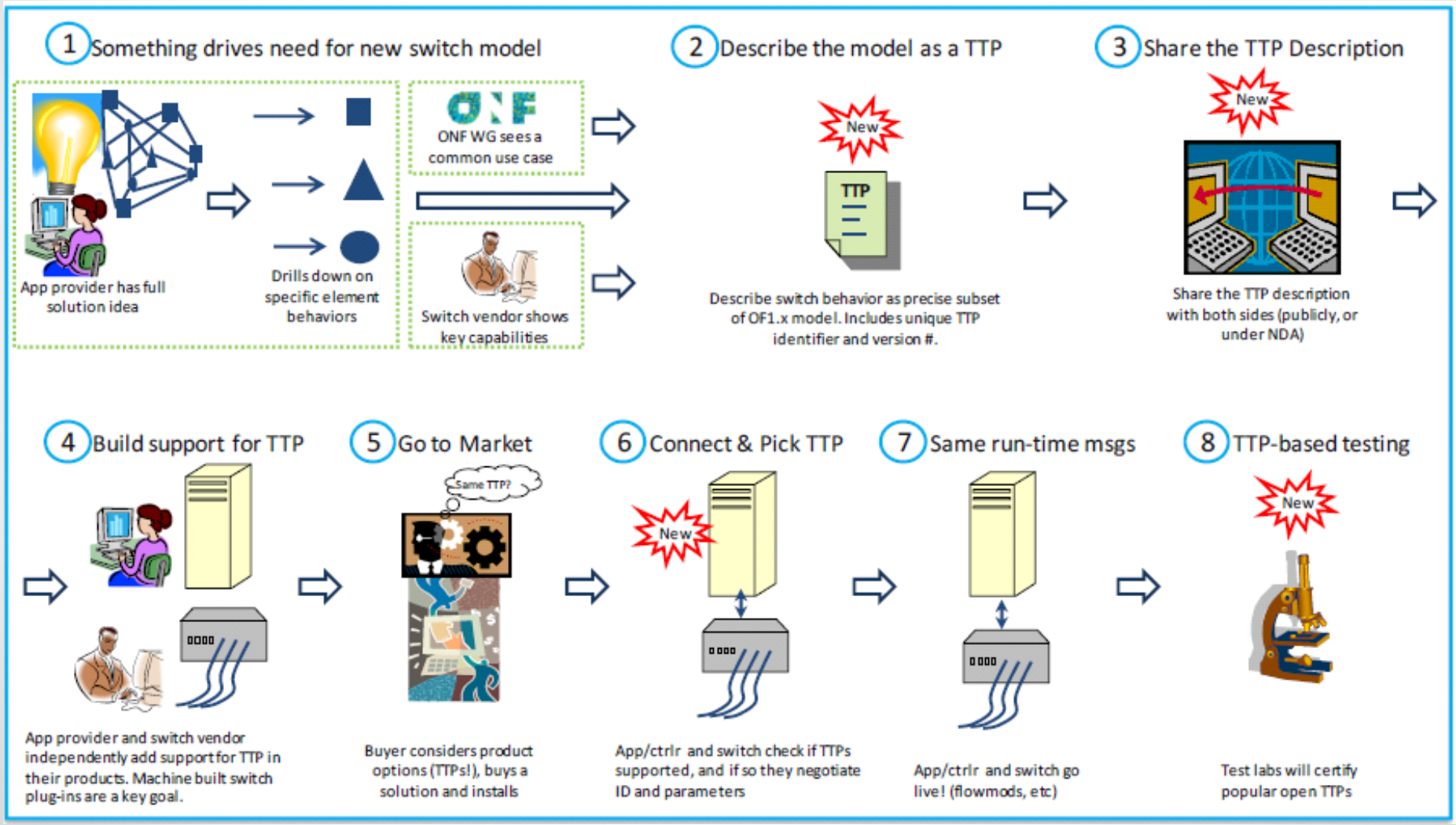
TTP framework



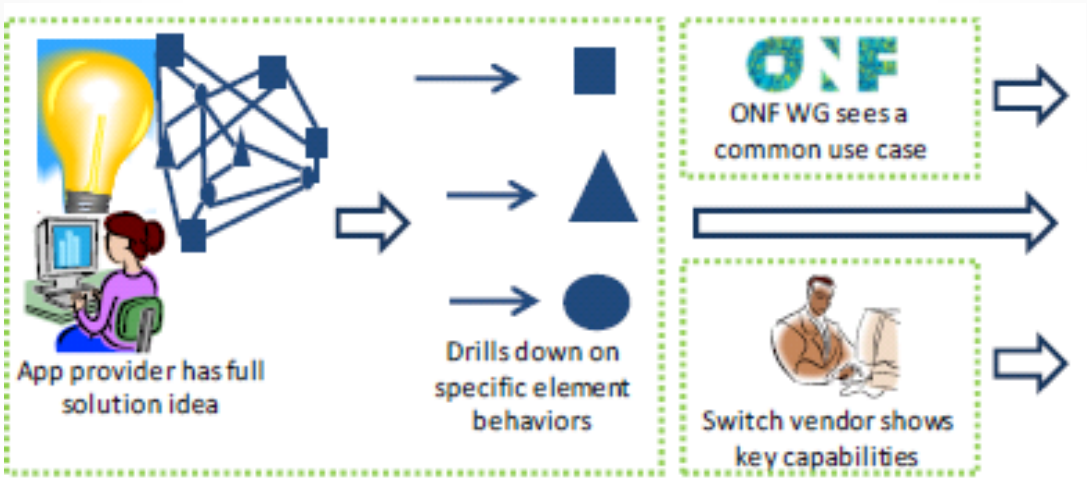
Before sending OpenFlow messages between controller and switch we need:

- Agree on specific TTP (configured a priory or negotiated)
- Multiply Flow Tables description (particularly device are ASIC based)
- JSON based syntaxes
- Includes a powerful mechanism Based on OF-CONFIG Protocol for synchronizing controller and switch TTP contexts

The basic Lifecycle of TTPs in an OpenFlow Network

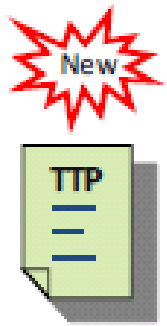


1. Something Prompts a New TTP



- The network architecture identifies the types of network elements (“logical switch” - defined as various shapes).
- Application providers (Vendors or network operators can conceive a various switch behaviors for solution architectures that they develop
- To get things started the ONF will define a small number of TTPs

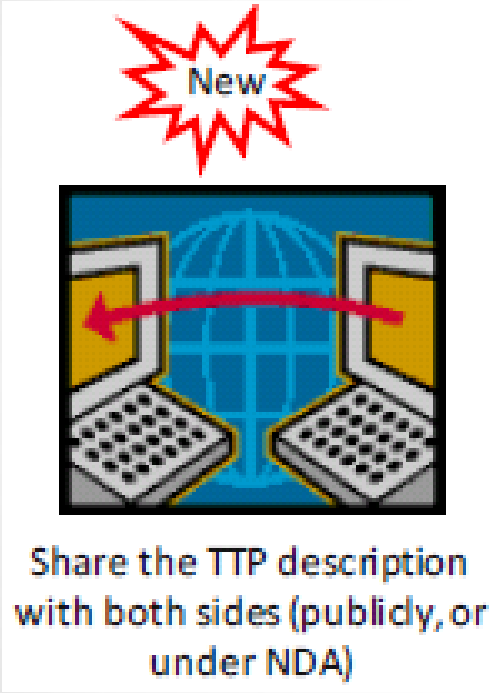
2. Describing a TTP



Describe switch behavior as precise subset of OF1.x model. Includes unique TTP identifier and version #.

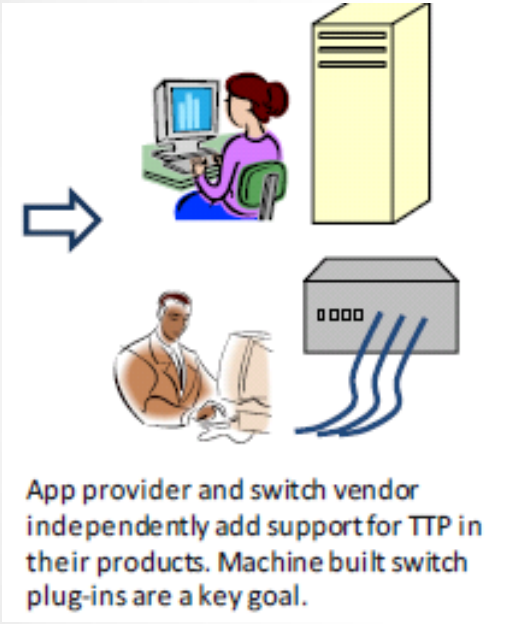
- Abstract switch model build on existing OpenFlow protocol version
- TTP is described as a proper subset of an OpenFLow multi-table logical switch
- Include Metadata (naming, authority, the Negotiable datapath model (NDM)type), name and version
- Includes a set of flow tables, flow mod messages types that must supported in specific flow tables, and the group entries required in the group table
- Define valid flow_mod and group_mod messages for a givvenlogical switch
- Effectively specifies “table graph” of the TTP

3. Sharing a TTP description



- ONF-developed TTPS
  - Once stable, fully public
    - Sharing will be accomplished by posting at the ONF website
    - Note that TTPs are not intended to be “standards”. Instead, TTPs are “common practices”, and ONF-developed TTPs will not hold any special distinction with respect to TTPs developed by others
- ONF Member-developed open TTPs
  - Application, controller and switch providers who are ONF members can develop a TTP and share it
    - Partnerships of members can also co-develop and share TTPs
    - Sharing of member TTPs will be supported on the ONF website, but can be accomplished in other way as well
- ONF Member-developed private TTPs
  - Market participants can also develop TTPs and keep them “under wraps”, either temporarily during development or for more extended periods. This might be handled in ways:
    - Fully hidden TTPS
    - Partially hidden TTPs (TTPs with Open IDs)
  - Private TTPs may, at the discretion of the controlling parties, be made open over time.

4. Build support for TTP



App provider and switch vendor independently add support for TTP in their products. Machine built switch plug-ins are a key goal.

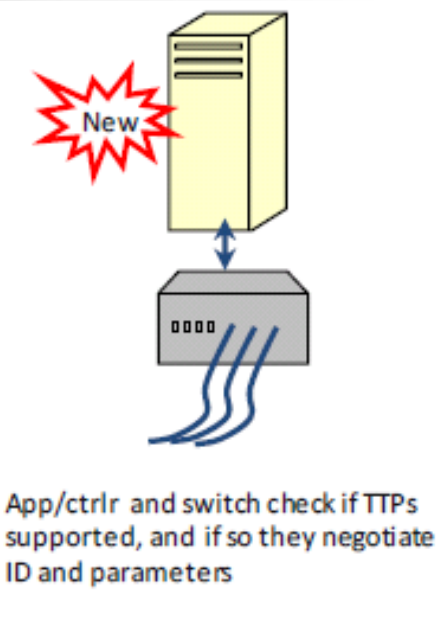
- Third party TTPs may be supported by both controller and switch in similar timeframe
- For TTPs provided by switch vendors and supported by their products, support can be added later by controller developers wishing to support this products in app.
- Hardware suppliers are adding support for a TTP :
  - Need to release typical hardware schedule
  - Be able to decouple TTP support from their main OS release, and offer shorter timescales for supporting new TTPs.

5. Go on Market

Buyer considers product options (TTPs!), buys a solution and installs

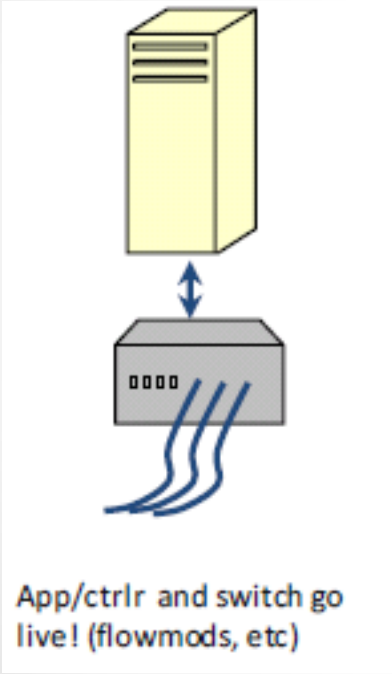
- Explicitly declaring support for applicable TTP gives customer confidence that controller from one vendor will interoperate with a switch from another vendor
- A very large network operator may define their own TTPs (own apps or controllers). May require support for this TTPs from their switch vendors

6. Connect & Pick TTP



- Discovery of available switch-supported TTPs, based on well-known TTP identifiers
- Simple activation of a desired TTP using default parameters
- An optional mechanism for the OFCP to negotiate the switch for preferred TTP parameters
  - Switches that support parameters negotiation must support an RPC extension defined within the NETCONF framework

7. Run-time messaging with TTP



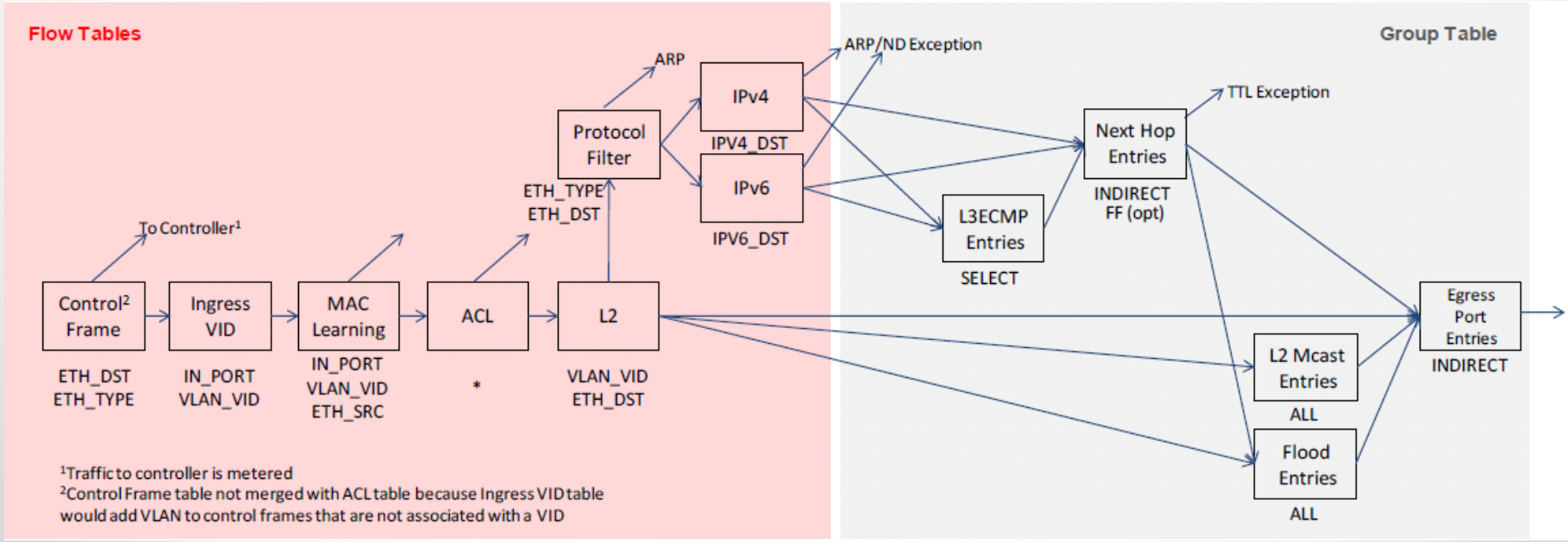
- Table Features messages will not be allowed to alter table attributes
- The switch may reject a message using a new error message if the controller attempts to exceed features defined by the TTP

8. TTP-based testing

- Product (both controller and switch) with built-in TTP awareness can be independently tested to assess conformance with specific TTPs.
  - TTPs are precise and unambiguous; if both ends conform to a TTP, interoperability is more easily achieved
- TTPs can be used as test profiles, even when the products lack build-in TTP awareness
  - The Testing and Interoperability WG has expressed a need for unambiguous test profiles
  - TTP Descriptions allow various market participants to define TTPs and, thus, test profiles
- TTP descriptions may in future include test information, to encourage self-testing of TTP conformance
  - As a TTP gains market traction, 3-rd party testing of that TTP will be increasingly likely
  - For new TTPs, self-testing provides a mechanism to build adoption until 3-rd party testing becomes available.

Example TTP Description: L2-L3-ACls

- Example provide both MAC forwarding (unicast, multicast) and IP forwarding (unicast only). Conforms to common standards and practices for MAC Bridging and IP forwarding



JSON sample

```
{
  "NDM_metadata": {
    "authority": "org.opennetworking.fawg",
    "type": "TTPv1",
    "name": "L2-L3-ACLs",
    "version": "1.0.0",
    "OF_protocol_version": "1.3.3",
    "doc": ["Example of a TTP supporting L2 (unicast, multicast, flooding), L3 (unicast only),",
            "and an ACL table."]
  },

  "security": {
    "doc": ["This TTP is not published for use by ONF. It is an example and for",
            "illustrative purposes only.",
            "If this TTP were published for use it would include",
            "guidance as to any security considerations in this doc member."]
  },

  "table_map": {
    "ControlFrame": 0,
    "IngressVLAN": 10,
    "MacLearning": 20,
    "ACL": 30,
    "L2": 40,
    "ProtoFilter": 50,
    "IPv4": 60,
    "IPv6": 80
  },
```

Table Type Patterns FAQ

If our device supports most of the messages of a TTP, can we claim support for the TTP?

- To legitimately claim support for a TTP, a switch must implement all non-optional functionality described by the TTP. As mentioned above, a TTP may describe some functionality that is optional. Support for the TTP can be fairly claimed even if the optional functions are not supported, though product documentation should make clear what is and is not supported. (For consistency in the marketplace, documentation of optional functionality support should refer to the OptFunc parameter values that are or are not supported. (See the OF-TTP specification and example TTPs for OptFunc details.)

Are TTPs intended to describe a particular chip pipeline?

- TTPs were conceived and developed as a way to describe “implementation independent” switch behavior (a use case), and thus TTP were intended to describe an abstract switch architecture rather than specific physical architecture. Such abstract architecture TTPs can be mapped onto different physical switch architectures. The expectation is that an abstract TTP that maps onto many physical devices will have an advantage in adoption. Application architects should therefore seek to develop such TTPs
- Nevertheless, TTPs can be written to cater to a particular physical device pipeline. Shortly before the TTP standard was approved, Broadcom introduced their OF-DPA effort, based on the TTP framework but focused on their switch architecture. (As of this writing, the TTP that represents OF-DPA capabilities is still in development, likely to complete within a few more weeks.)
- If a particular switch vendor’s TTP gains an adoption advantage, one possible outcome is that other vendors may be motivated to support that TTP, which would represent a form of market-driven functional convergence at the device level. In this way, TTPs can support an evolution toward a long-promised SDN reality.

Are TTP files sent over the wire?

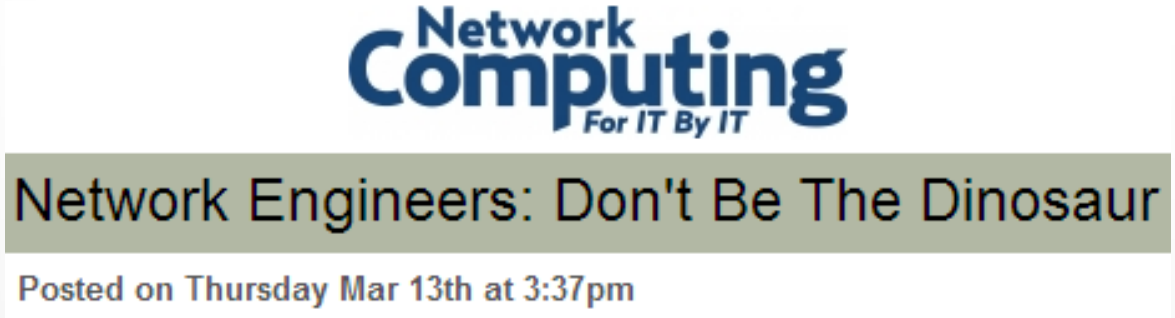
- No. The current TTP framework does not describe any situation where OpenFlow products would pass TTP descriptions "over the wire". However, there will likely be scenarios in the future where such transfer will become interesting. For now, experimenter extensions to the OFSwitch protocol offer one way to implement the capability for providers that have interest in features that could benefit from over-the-wire TTP transfer.

Do a controller and a switch need to synchronize with each other to use TTPs?

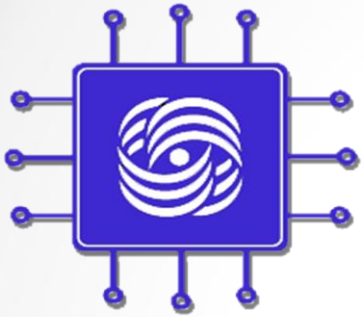
- When TTP support is integrated into products (controllers and switches), then it is necessary to synchronize the two endpoints to enable TTP-related functionality. This can be done by configuration or during OF Switch initialization. At this time, the config-based approach would either use the OF-Config-based NDM Sync mechanism (see the NDM sync spec) or some proprietary (out-of-scope) mechanism of configuration such as CLI commands at both ends. The OF-Switch-based mechanism would use an OF-Switch extension mechanism that is, at the time this is written, under development as an experimenter extension by the Forwarding Abstractions WG.

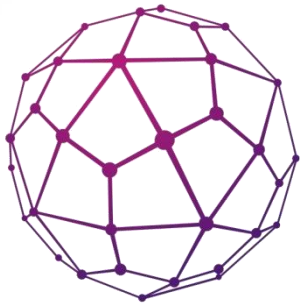
Заключение

- OF-DPA огромный шаг Broadcom на встречу SDN/OpenFlow
  - Описывает pipeline чипсета в терминах OpenFlow
- NETCONF/YANG хитрый (но логичный шаг) CISCO к управлению своим оборудованием



http://arccn.ru/
ashalimov@lvk.cs.msu.su
@alex_shali
@arccnnews





ЦЕНТР
ПРИКЛАДНЫХ
ИССЛЕДОВАНИЙ
КОМПЬЮТЕРНЫХ
СЕТЕЙ

Спасибо за внимание!