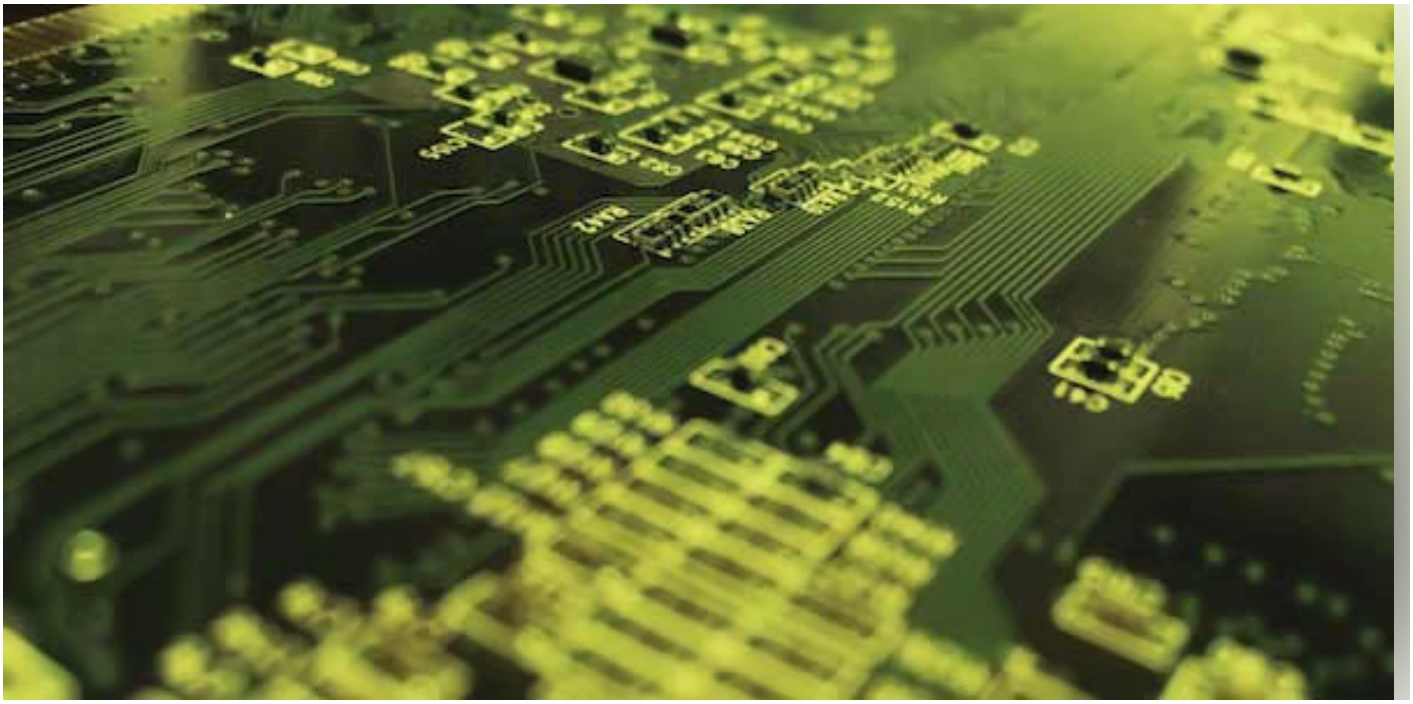
Articles / **Contributed**

TCAMs and OpenFlow - What Every SDN Practitioner Must Know

Brent Salisbury July 2, 2012 1:00 AM



Trying to negotiate SDN-enabled hardware today is not an easy thing to do. This is primarily due to current networking hardware not designed for a decoupled control plane. Yet, the release of the OpenFlow v1.0 Spec coupled with customer demand has driven vendors to provide OpenFlow-networking gear. Using existing off the shelf hardware, manufacturers have taken their network operating systems (NOS) and interrupted the normal pipeline processing to fork selected packet flows into the OpenFlow pipeline. In order to achieve this level of flexibility on such refined, purpose-built hardware switches, it requires flexibility and performance in its forwarding tables. The quick fix for the early stages of experimentation and proofing on real world hardware is leveraging Ternary Content Addressable Memory (TCAM).

Why have we heard so much about TCAMs lately?

Today the supported protocol used to instantiate a flow into vendor switches who support SDN is OpenFlow v1.0. Some manufacturers have begun supporting v1.1 and greater, but none of that is shipping in GA. Along that thread, most vendors appear to be skipping v1.1 all together for v1.2 and v1.3.

Generally speaking, we can look at hardware spec sheets with the layers and protocols we have in our industry paradigm today and understand capabilities of hardware and how they are weaved or wedged into our networks. Though we are still left programming essentially firmware on each device it is just the reality of the industry for now.

Highlights of the OpenFlow Release Specs

- Version 1.1 – Multi-table pipeline processing, MPLS, QinQ.
- Version 1.2 – IPv6 and additional extensibility.
- Version 1.3 – (Release Pending) QOS additions among others.

Simply put, the first packet of a flow checks the OpenFlow forwarding tables for a match, if one is not found, it punts to the controller and the controller inserts a flow entry in the switches forwarding table. *if match; then forward via flow table, else; forward to controller*. With OpenFlow v1.0, early hardware adopters use Ternary Content Addressable Memory (TCAM) to insert a flow from a controller into the forwarding tables or forwarding information base (FIB) for subsequent flow lookups of a flow. That does not give much room for hardware differentiation other than having external TCAM on board for more flow tables in TCAM.

TCAM and OpenFlow

TCAM naturally lends itself to being great for flow instantiation from an SDN controller to inject forwarding entries with a great deal of flexibility across various Ethernet and IP header fields e.g. 10-tuple OpenFlow v1.0 header fields *Ingress Port, Ethec Src, Ether Dst, Ether Type, Vlan ID, IP Dst, IP Src, TCP Dst, TCP Src, IP Proto*.

Most vendors have TCAM in place for match things such as IP access-lists. That TCAM using software has been modified through either a HAL or some encoder like software algorithm to expand the width of the TCAM for OpenFlow lookups.

Ingress Port	Ethec Src	Ether Dst	Ether Type	Vlan ID	IP Dst	IP Src	TCP Dst	TCP Src	IP Proto

Figure 1. 12-Tuples of headers.

Content Addressable Memory

CAM is a unique memory that can do memory lookups in one clock cycle and in a parallel fashion looking at multiple fields at once in a lookup. This is the most common way to accommodate hardware search ASICs. For programmers, CAM can be described as associative array logic.

- Two type typical hardware lookups are Binary CAM lookups (BCAM) and Ternary CAM lookups (TCAM)

Binary CAM – As the word binary implies, is a binary lookup at returns either a 1 or 0. As exemplified in Figure 2, a mac address in an Ethernet frame comes into a switch, the switch looks in its mac address table and either finds that make address or doesn't, 1 or 0.

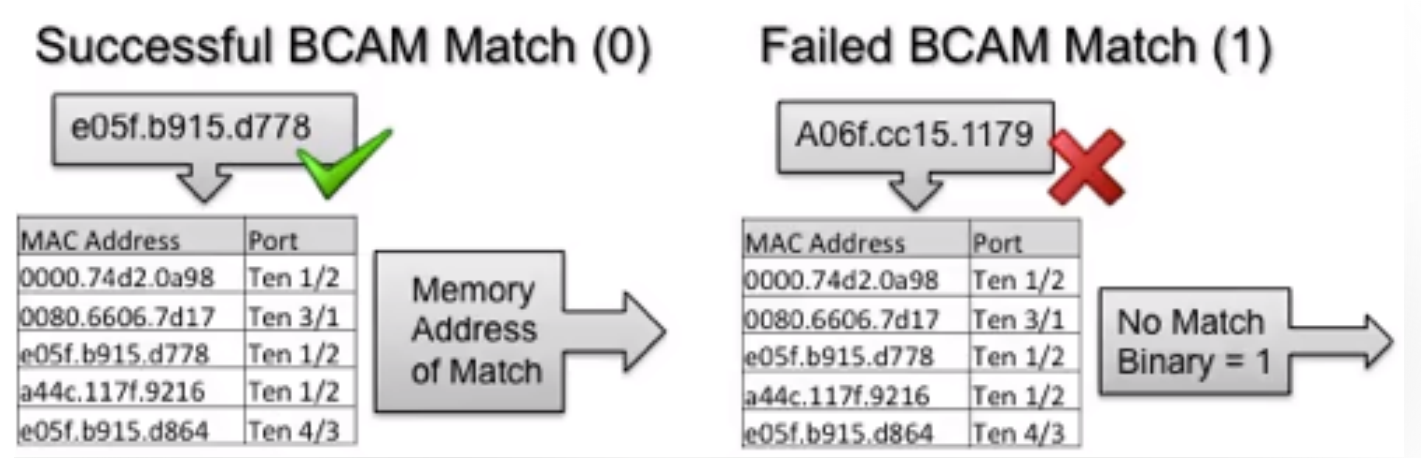


Figure 2. Binary CAM operation example

Ternary CAM – TCAM adds a 3rd additional component as the name 'ternary' suggest. TCAM lookups match explicit 1 and 0 but also have a 'don't care' bit. You will also hear this referred to with regard to OpenFlow as a wildcard bit often depicted to in diagrams with a '*'. TCAM can have multiple matches and determine a best match. This is necessary since CIDR lookups need to support a longest prefix match. Example 192.168.1.7/32 would match on 192.168.1.0/24 and 192.168.1.0/25. The closest match to 192.168.1.7/32 is 192.168.1.0/25 and would be chosen.

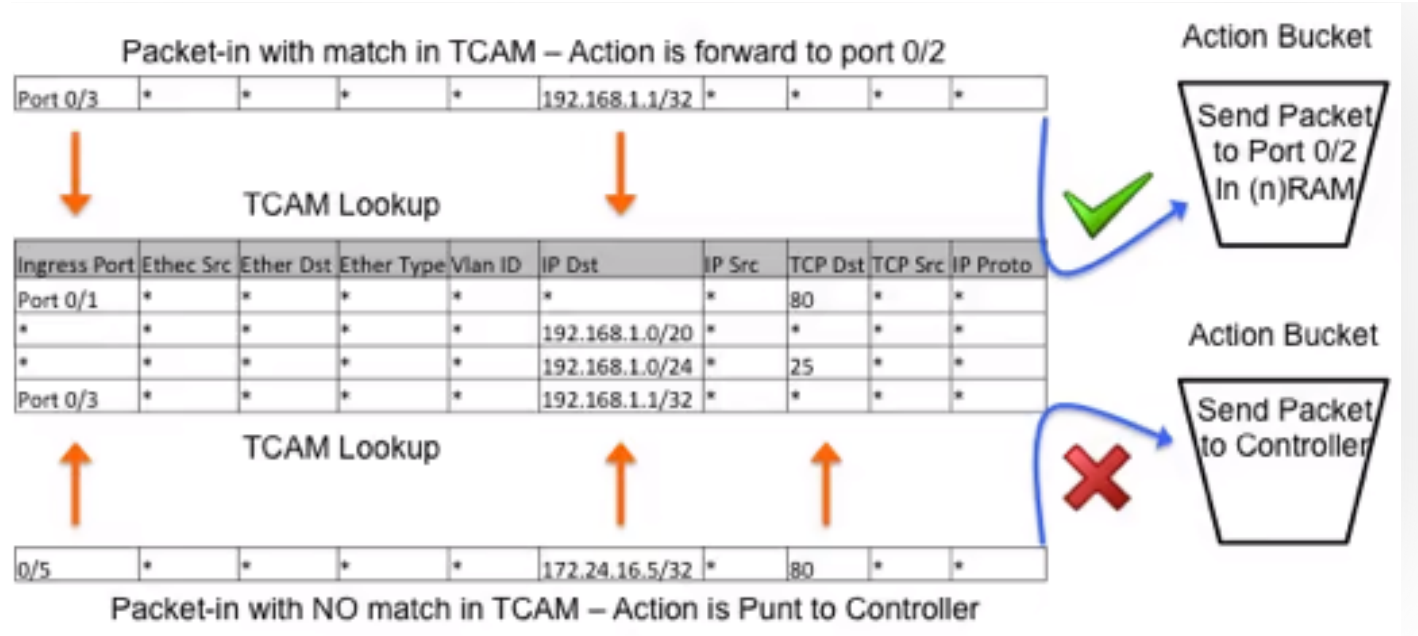


Figure 3. Ternary CAM OpenFlow operation example

TCAM Limitations

TCAM is not the cure all for hardware search lookups. TCAM is power hungry, expensive and takes up quite a bit of silicon space. It would not be unusual for it to be the most expensive component on commodity switches. Many vendors use a blend of BCAM memory, SRAM, NPUs and software algorithms to perform ternary lookups (see MX trio chipset and EzChip NPUs in ASRs, Trie lookups). There are techniques depending on how many explicit matches are made across n-tuples to mask wildcard fields in a TCAM entry. You may hear from vendors that flow table capacity will depend on your desire to match a given number of tuples. If you are looking for exact matches on all tuples in every flow it may affect the total number of supported flows while matching a couple tuples in flows with the ability to mask more fields. That said, there are incredibly clever software algorithms that can even overcome full flow match limitations to some degree.

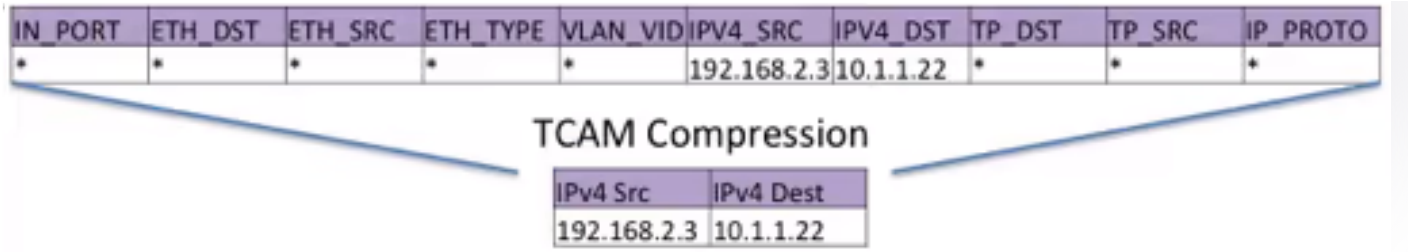


Figure 4. Example of TCAM compression to preserve TCAM space.

Looking Ahead

Starting with OpenFlow v1.1 and beyond rather than having one flow table there is a pipeline made up of multiple flow tables. While this does add complexity to the standard it also allows for more flexibility in leveraging different hardware components rather than just raw TCAM to do associative array logic. Brocade has a slick implementation of protected and unprotected modes that can optimize for hybrid deployments or migration to balance the normal FIB and SDN tables. This is important for my personal use

cases in research and education networks, to allow researchers to have research networks consolidated on the same network substrate as traditional production services.

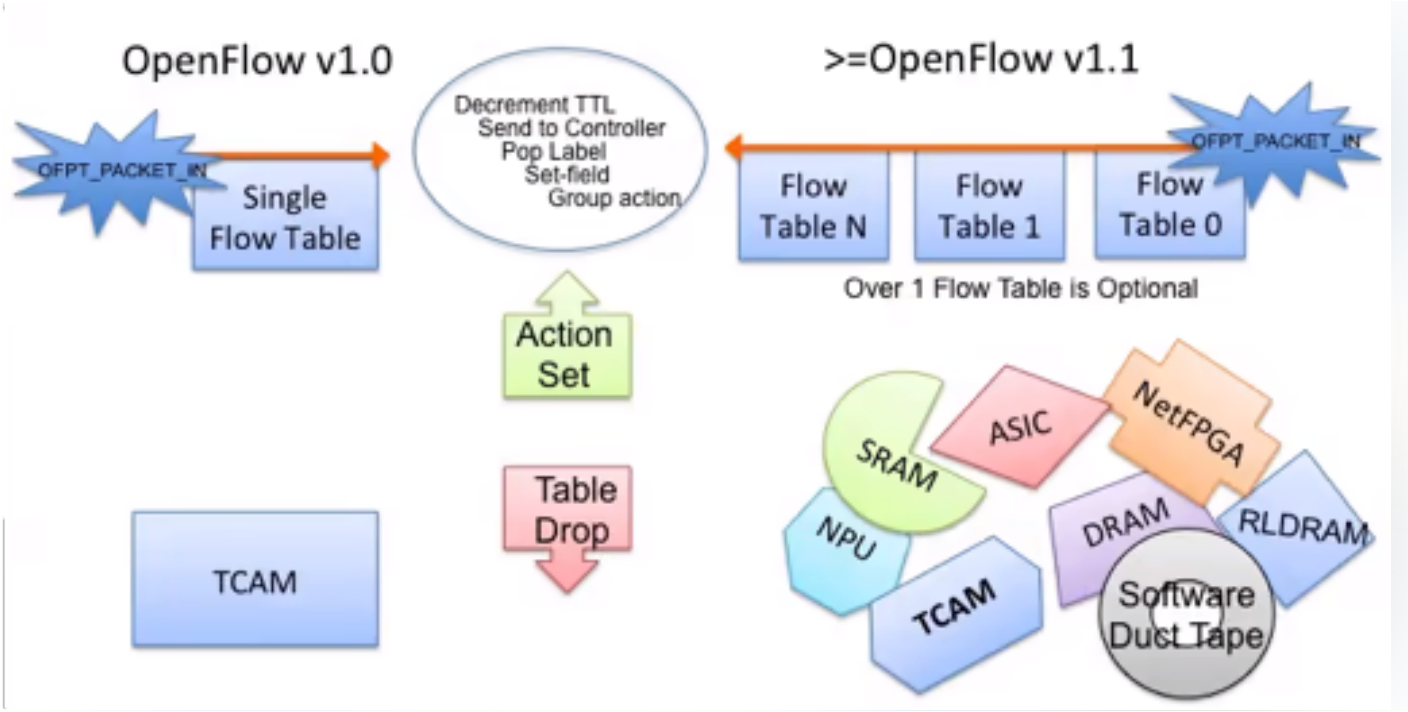


Figure 5. There will be new combinations of hardware and software for SDN lookups coming.

Nick McKeown put it quite eloquently in a meeting in February of 2010 in the weekly OpenFlow meeting notes. The topic multiple flow tables was being discussed. “We want to create a dynamic where we have a very good base set of vendor-agnostic instructions. On the other hand, we need to give room for switch/chip vendors to differentiate.”

This ability to distinguish hardware platforms may the balancing act to squeeze performance out of the hardware market for the next few generations of network devices as the industry dips its toes into SDN while juggling normal forwarding tables, and even more complicated, interactions between the two pipelines. When choosing hardware for proof of future proof of concepts or reading new spec sheets with application towards SDN the flow tables and ACL like tuple matching will be relatively important when trying to pre-determine performance and scale. Software will be just as important on the southbound interface as it is on the north, but hopefully over the next few years, we can get back to being blissfully unaware of what is happening below the UI when implementing policy and architectures.