OVS Configuration Guide

March, 2015

Version: 1

www.pica8.com

Pica8, Inc.
1032 Elwell Court, Suite 105
Palo Alto, CA. 94303
+1 (650) 614-5838

sales@pica8.com
support@pica8.com

Contents

Introduction

- Supported OpenFlow Protocol 1.4 Features
- PicOS OVS OF-1.3 Support

Basic configuration in OVS mode

Troubleshooting OVS

Configuring Open vSwitch

- Creating a Bridge
- Connecting to a Controller
- Setting the Port Link Speed
- Configuring a Trunk Port
- 40G Changes to 4*10G in OVS
- Configuring sFlow v5
- Configuring NetFlow
- Configuring Port Mirroring
- Configuring IPv4 OpenFlow
- Configure GRE Tunneling
- Configuring MPLS
- Configuring LAG and LACP
- Creating a Group Table
- Configuring ECMP
- Class of Service Mapping for QoS
- Configuring QoS Queue
- Configuring OpenFlow Meter
- Configuring QinQ
- Configuring OpenFlow Provider Backbone Bridge
- Configuring OpenFlow Loopback
- Enabling Loopback Interface
- Optimizing TCAM Usage
- Configuring Layer 2 over GRE on 5101 and 5401 Switches
- Configuring VXLAN
- Configuring Multi-Table
- Configuring Network Address Translation
- ASIC Limitation
- Configuring CFM
- OVS Configuration File

OVS Web User Interface

- Login Interface
- Monitor
- Adding a Bridge
- Add a Port
- Add GRE Port
- Add Group Table
- Add or Edit a Controller
- Edit Flow Tables
- Edit Lag Interface

Examples and Topologies

- 802.1Q VLAN
- ECMP
- GRE Tunnel
- MPLS Network
- Multiple Virtual Bridges
- SSL Connection to Controller

Create SSL connection with controller

Introduction

Pica8's operating system, PicOS, leverages Open vSwitch (OVS) a production quality, multi-layer virtual switch licensed under the open source Apache 2.0 license. OVS runs as a process within PicOS.

The OpenFlow (OF) protocol is driven by the Open Networking Foundation (ONF), a leader in software-defined networking (SDN). The OpenFlow protocol governs three essential components of SDN: an OpenFlow physical switch, an OpenFlow virtual switch to manage virtual machines, and an OpenFlow controller to organize all network pieces.

PicOS supports features in OpenFlow release 1.3 and 1.4. Details of the supported features are outlined in Table 1 and Table 2. The following web sites provide more detailed information on Open vSwitch and the OpenFlow protocol.

Open vSwitch http://openvswitch.org/
OpenFlow https://www.opennetworking.org/sdn-resources/onf-specifications/openflow

PicOS can run in 2 different modes of operation:

- Open vSwitch (OVS) mode - In this mode PicOS is dedicated and optimized for Openflow applications
- Layer 2 / Layer 3 (L2/L3) mode - In this mode, PicOS can run both switching and routing protocols and Openflow applications

In OVS mode, L2/L3 daemons are not running; the system is fully dedicated to Openflow and OVS. In L2/L3 mode, L2/L3 daemons are running, OVS can also be activated if Crossflow is activated.This chapter suppose that the OVS mode is activated in PicOS. Please see PicOS Mode Selection to modify the PicOS mode.

Supported OpenFlow Protocol 1.4 Features

Table 1 contains the OpenFlow protocol 1.4 features supported by PicOS. For clarity, the feature names are identical to the feature names found in the OpenFlow Switch Specification version 1.4.

Table 1 OpenFlow Protocol 1.4 Features

Pica8 OpenFlow V1.4 Compliance Matrix			
Chapter	**Tile**	**Features**	**Detail Feature Specification**
2	Switch Components	NA	
		flow table	

		group table	all, indirect, select, fast_failover gr table are all supported
		add/update/delete flow entries	
		match fields	
		counters	
		a set of instructions	
4	OpenFlow Ports	NA	
4.1	OpenFlow Ports	see 4.2-4.5	
4.2	Standard Ports	see 4.2-4.5	
4.3	Physical Ports	NA	
		ingress	PicOS only support it as matching
		output	
		hardware interface	
		groups	
		port counters	
4.4	Logical Ports	NA	The OpenFlow logical ports are sv defined ports that don't correspon directly to a hardware interface of switch
		map to various physical ports	
		LAG	
		tunnel (GRE)	
		lookback interface	
		ingress	

		output	
		groups	
4.5	Reserved Ports	NA	
		all	Represents all ports the switch ca for forwarding a specific packet, C be used only as an output port
		controller	Represents the control channel w OpenFlow controller, Can be used as an ingress port or as a output port
		table	Represents the start of the OpenF pipeline
		in_port	Represents the packet ingress po be used only as an output port,se packet out through its ingress por
		any	Special value used in some Open commands when no port is specified,Can neither be used as ingress port nor as an output port
		local	Represents the switch's local netv stack and its management stack.
		normal	Represents the traditional non-OpenFlow pipeline of the swi ,Can be used only as an output p processes the packet using the n pipeline
		flood	Represents flooding using the nor pipeline of the switch,Canbe used as an output port
5	OpenFlow Tables	NA	

5.1	Pipeline Processing		
		openflow-only	all packets are processed by the OpenFlow pipeline
		openflow-hybrid	OpenFlow operation and normal Ethernet switching operation
			L2 Ethernet switching, L3 routing routing, IPv6 routing...),ACL and C processing
			VLAN isolation
			a classification mechanism outsid OpenFlow that routes traffic to eit OpenFlow pipeline or the normal pipeline
			vlan tag or input port whther to pr the packet using whitch pipeline
			normal and flood
		multiple flow tables, each flow table containing multiple flow entries	
		sequentiallly numbered, start at 0	
		goto instruction	
		go forward and not buckward	
		last table can not include goto instruction	
		table miss	
5.2	Flow Table	NA	

		match fields	to match against packets.
		priority	matching precedence of the flow e
		counters	updated when packets are matche
		instructions	to modify the action set or pipeline processing
		timeouts	maximum amount of time or idle t before flow is expired by the switc
		cookie	opaque data value chosen by the controller
		wildcards all fields and priority equal 0 is table-miss	
5.3	Matching	NA	
		ingress port	
		metadata fields	
		apply-actions	
		any	
		highest priprity matches packets be select	
		couters update and instruction applied	
		OFPFF_CHECK_OVERLAP	
		OFPC_FRAG_REASM	
		behavior when a switch receives a corrupted packet	
5.4	Table-miss	NA	

		every flow table support table-miss	
		send packets to controller	
		drop packets	
		direct packets to a subsequent table	
		wildcard all match fields	
		priority= 0	
		not exist by default	
		add or remove by controller at any time	
		may expire	
		match packets unmatched by others	
		instructions applied	
		packet-in reason is table-miss	
		packets unmatchd are dropped is not exist table-miss	
5.5	Flow Removal	NA	
			is run by the switch independantly controller and is based on the stat configuration of flow entries
			A non-zero hard_timeout field cau the flow entry to be removed after given number of seconds, regardl how many packets it has matched

			A non-zero idle_timeout field caus flow entry to be removed when it l matched no packets in the given number of seconds
			The switch must implement flow e and remove flow entries from the table when one of their timeout is exceeded
		OFPFF_SEND_FLOW_REM flag	When a flow entry is removed, the switch must check the flow entry's OFPFF_SEND_FLOW_REM flag. flag is set, the switch must send a removed message to the controlle
			Each flow removed message cont complete description of the flow e the reason for removal (expiry or delete), the flow entry duration at time of removal, and the flow stati time of removal
		eviction	Flow entries may be evicted from flow tables when the switch needs reclaim resources
5.6	Group Table	NA	
		group identifier	
		group type	
		counters	
		action buckets	
5.6.1	Group Types	all	used for multicast or broadcast
			effectively cloned for each bucket

			process for each bucket
			direct out the ingress, packet is dr
			output action to OFPP_IN_PORT
		select	process by a single bucket
			switch-computed selection algorit
			bucket weight
			forward to live ports
		indirect	support only a single bucket
			multiple ow entries or groups to point to a common group identi er
			supporting faster, more ecient convergence
			effectively identical to an all group one bucket
		fast failover	execute the first live bucket
			associated with a port and/or grou
			change forwarding without reques controller
			no bucket live, packet dropped
5.7	Meter Table		

		meter entries	
		per-flow meters	
		rate-limit	
		combine with per-port queue	
		measure and control packet rate	
		attached to flow entries	
		in flow instruction set	
		multiple meters in the same table	
		multiple meters on the same set of packets	
		meter identifier	
5.7.1		Meter Bands	
		one band	
		more meter bands	
		the rate band applies and the way packets be process	
		proccessed by a single meter band based on the current measured meter tate	
		configure rate lower than current rate	
		no meter band applied if current rate lower than specified rate	

		band type	
		rate	
		couters	
5.8	Counters	NA	
	Per Table Counters	Reference count (active entries)	32 bits
		Packet Lookups	64 bits
		Packet Matches	64 bits
	Per Flow Counters	Received Packets	64 bits
		Received Bytes	64 bits
		Duration (seconds)	32 bits
		Duration (nanoseconds)	32 bits
	Per Port Counters	Received Packets	64 bits
		Transmitted Packets	64 bits
		Received Bytes	64 bits
		Transmitted Bytes	64 bits
		Receive Drops	64 bits
		Transmit Drops	64 bits
		Receive Errors	64 bits
		Transmit Errors	64 bits
		Receive Frame Alignment Errors	64 bits
		Receive Overrun Errors	64 bits

		Receive CRC Errors	64 bits
		Collisions	64 bits
	Per Queue Counters	Transmit Packets	64 bits
		Transmit Bytes	64 bits
		Transmit Overrun Errors	64 bits
	Per Group Counters	Reference Count (flow entries)	32 bits
		Packet Count	64 bits
		Byte Count	64 bits
	Per Bucket Counters	Packet Count	64 bits
		Byte Count	64 bits
5.9	Instructions		
		The controller can query the switch about which of the “Optional Instruction” it supports	
		Apply-Actions action(s)	
		Clear-Actions	
		Write-Actions action(s)	
		Write-Metadata metadata / mask	
		Goto-Table next-table-id	

		Clear-Actions instruction is executed before the Write-Actions instruction	
		Goto-Table is executed last	
		reject a flow entry if it is unable to execute the instructions & return an unsupported flow error	
5.10	Action Set		
		action set is associated with each packet	
		This set is empty by default	
		action set is carried between flow tables	
		When the instruction set of a flow entry does not contain a Goto-Table instruction, pipeline processing stops and the actions in the action set of the packet are executed	
		action set contains a maximum of one action of each type	
		The actions in an action set are applied in the order specified below	
		1. copy TTL inwards	
		2. pop	
		3. push	
		4. copy TTL outwards	

		5. decrement TTL	
		6. set	
		7. qos	
		8. group	if a group action is specified, appl actions of the relevant group buck in the order specified by this list
		9. output	if no group action is specified, forv the packet on the port specified by output action. The output action ir action set is executed last
			If both an output action and a grou action are specified in an action s output action is ignored and the g action takes precedence
			If no output action and no group a were specified in an action set, th packet is dropped
			The execution of groups is recurs the switch supports it; a group buc may specify another group, in whi case the execution of actions trav all the groups specified by the gro configuration
5.11	Action list		
		Apply-Actions instruction and the Packet-out message include an action list	The actions of an action list are executed in the order specified by list, and are applied immediately t packet
			The effect of those actions is cum
			If the action list contains an outpu action, a copy of the packet is forv in its current state to the desired p

			If the list contains a group actions copy of the packet in its current st processed by the relevant group b
5.12	Actions		
		Output	support forwarding to physical por switch-defined logical ports and th required reserved ports
		Set-Queue	The set-queue action sets the que for a packet and is used to provid Quality-of-Service (QoS) support
		Drop	
		Group	
		Push-Tag/Pop-Tag	order of header fields - Ethernet, V MPLS, ARP/IP, TCP/UDP/SCTP (IP-only)
		Push VLAN header	Push a new VLAN header onto th packet. The Ethertype is used as Ethertype for the tag. Only Etherty 0x8100 and 0x88a8 should be us
		Pop VLAN header	Pop the outer-most VLAN header the packet
		Push MPLS header	Push a new MPLS shim header o packet. Only Ethertype 0x8847 and 0x8848 sh used.
		Pop MPLS header	Pop the outer-most MPLS tag or s header from the packet.
		Push PBB header	
		Pop PBB header	
		Set-Field	

		Set VLAN ID	
		Strip VLAN ID	
		Change-TTL	modify the values of the IPv4 TTL Hop Limit or MPLS TTL in the pac
			If it is supported, applied to the outermost-possible header
		Set MPLS TTL	8 bits: New MPLS TTL, Replace t existing MPLS TTL. Only applies packets with an existing MPLS sh header
		Decrement MPLS TTL	Decrement the MPLS TTL. Only a to packets with an existing MPLS shim header
		Set IP TTL	Replace the existing IPv4 TTL or Hop Limit and update the IP checksum. Only applies to and IPv6 packets.
		Decrement IP TTL	Decrement the IPv4 TTL or IPv6 H Limit field and update the IP checksum. Only applies to and IPv6 packets
		Copy TTL outwards	Copy the TTL from next-to-outerm outermost header with TTL. Copy can be IP-to-IP, MPLS-to-MPLS, or IP-to-MPLS
		Copy TTL inwards	Copy the TTL from outermost to next-to-outermost header with TTL. Copy can be IP-to-IP, MPLS-to-MPLS, or MPLS-to-IP
5.12.1	Default values for field on push		

		Field values for all fields specified in Table 6 should be copied from existing outer headers to new outer headers	VLAN ID VLAN ID
		New fields listed in Table 6 without corresponding existing fields should be set to zero	VLAN priority VLAN priority
		Fields in new headers may be overridden by specifying a “set” action for the appropriate field(s) after the push operation	MPLS label MPLS label
		Fields in new headers may be overridden by specifying a “set” action for the appropriate field(s) after the push operation	PBB label PBB label
6	OpenFlow Channel		
6.1	OpenFlow Protocol Overview	The OpenFlow protocol supports three message types, controller-to-switch, asynchronous, and sym-metric, each with multiple sub-types	
6.1.1		Controller-to-Switch	Controller/switch messages are in by the controller and may or may require a response from the switc
6.1.2		Asynchronous	Asynchronous messages are sen without a controller soliciting them a switch
			Switches send asynchronous messages to contr to denote a packet arrival or switc change

6.1.3		Symmetric	Symmetric messages are sent wit solicitation, in either direction., inc Hello, Echo, Error, Experimenter message
6.2	Message Handling	The OpenFlow protocol provides reliable message delivery and processing, but does not automatically provide acknowledgements or ensure ordered message processing.	
6.3	OpenFlow Channel Connections	The OpenFlow channel is used to exchange OpenFlow message between an OpenFlow switch and an OpenFlow controller	
6.3.1		Connection Setup	The switch must be able to establ communication with a controller a user-configurable IP address, usir either a user-specified transport p the default transport port
6.3.2		Connection Interruption	In the case that a switch loses cor with all controllers the switch mus immediately enter either \fail secu mode" or \fail standalone mode", depend upon the switch implementation a configuration
6.3.3		Encryption	The switch and controller may communicate through a TLS conr
6.3.4		Multiple Controllers	The switch may establish communication with a single contr or may establish communication v multiple controllers
6.3.5		Auxiliary Connections	The OpenFlow channel may also composed of a main connection a multiple auxiliary connections

6.4	Flow Table Modification Messages	Flow table modification messages are used to add, modify, delete flow	
6.5	Flow Table Synchronisation	A flow table may be synchronised with another flow table. with Flow Table Synchronisation	
6.6	Group Table Modification Messages	Action of group (including add, modify, delete) can be done by Group table modification messages	
6.7	Meter Modification Messages	Action of meter (including add, modify, delete) can be done by Meter modification messages	
6.8	Bundle Messages		
6.8.1		Bundle overview	A bundle is a sequence of OpenF modification requests from the co that is applied as a single OpenFl operation
6.8.2		Bundle example usage	
6.8.3		Bundle error processing	The OpenFlow messages part of bundle must be pre-validated befc they are stored in the bundle
6.8.4		Bundle atomic modifications	Committing the bundle must be controller atomic,
6.8.5		Bundle parallelism	The switch must support exchang echo request and echo reply mes during the creation and population of the bundle, the swit must reply to an echo request witl waiting for the end of the bundle

7	The OpenFlow Protocol		
7.1	OpenFlow Header	Each openflow message begins with the OpenFlow header	
7.1.1		Padding	Most OpenFlow messages contai padding fields
7.2	Common Structures		
7.2.1		Port Structures	The switch may define physical a logical ports
7.2.1.1		Port Description Structures, The physical ports, switch-defined logical ports, and the OFPP_LOCAL reserved port	ports includes OFPP_IN_PORT,OFPP_TABLE, OFPP_NORMAL, OFPP_FLOOD OFPP_ALL, OFPP_CONTROLLER, OFPP_LOCAL,OFPP_ANY
7.2.1.2		Port Description Properties A property definition contains the property type, length, and any associated data	associated date includes curr, advertised, supported, peer, each consists of speed, duplexity
7.2.2		Flow Match Structures	An OpenFlow match is composed ow match header and a sequence zero or more ow match fields
7.2.2.1		Flow Match Header, Fields to match against flows	The ow match header is described by ofp_match structure, The type fiel to OFPMT_OXM and length field to the actual length of ofp_match structure including all match fields. The pay the OpenFlow match is a set of O Flow match fields.

7.2.2.2		Flow Match Field Structures	The ow match fields are described usi OpenFlow Extensible Match (OXM format, which is a compact type-length-value (TLV) format
7.2.2.3		OXM classes	The match types are structured us OXM match classes, The OpenFl specification distinguishes two typ OXM match classes, ONF membe classes and ONF reserved classe diferentiated by their high order bi
7.2.2.4		Flow Matching	A zero-length OpenFlow match (o with no OXM TLVs) matches ever packet Match fields that should be wildcarded are omitted the OpenFlow match.
7.2.2.5		Flow Match Field Masking	The masks are defined such that a 0 i given bit position indicates a \don' match for the same bit in the corresponding field, wherea means match the bit exactly
7.2.2.6		Flow Match Field Prerequisite	In general, matching header fields protocol can only be done if the OpenFlow match explitly matches the corresponding protocol.
7.2.2.7		Flow Match Fields	match fields contains OFPXMT_OFB_IN_PORT, OFPXMT_OFB_IN_PHY_PORT,
		OXM_OF_IN_PORT	/* Switch input port. */
		OXM_OF_IN_PHY_PORT	/* Switch physical input port. */
		OXM_OF_METADATA	/* Metadata passed between table
		OXM_OF_ETH_DST	/* Ethernet destination address. */

		OXM_OF_ETH_SRC	/* Ethernet source address. */
		OXM_OF_ETH_TYPE	/* Ethernet frame type. */
		OXM_OF_VLAN_VID	/* VLAN id. */
		OXM_OF_VLAN_PCP	/* VLAN priority. */
		OXM_OF_IP_DSCP	/* IP DSCP (6 bits in ToS field). */
		OXM_OF_IP_ECN	/* IP ECN (2 bits in ToS field). */
		OXM_OF_IP_PROTO	/* IP protocol. */
		OXM_OF_IPV4_SRC	/* IPv4 source address. */
		OXM_OF_IPV4_DST	/* IPv4 destination address. */
		OXM_OF_TCP_SRC	/* TCP source port. */
		OXM_OF_TCP_DST	/* TCP destination port. */
		OXM_OF_UDP_SRC	/* UDP source port. */
		OXM_OF_UDP_DST	/* UDP destination port. */
		OXM_OF_SCTP_SRC	/* SCTP source port. */
		OXM_OF_SCTP_DST	/* SCTP destination port. */
		OXM_OF_ICMPV4_TYPE	/* ICMP type. */
		OXM_OF_ICMPV4_CODE	/* ICMP code. */
		OXM_OF_ARP_OP	/* ARP opcode. */
		OXM_OF_ARP_SPA	/* ARP source IPv4 address. */
		OXM_OF_ARP_TPA	/* ARP target IPv4 address. */
		OXM_OF_ARP_SHA	/* ARP source hardware address.
		OXM_OF_ARP_THA	/* ARP target hardware address.

		OXM_OF_IPV6_SRC	/* IPv6 source address. */
		OXM_OF_IPV6_DST	/* IPv6 destination address. */
		OXM_OF_IPV6_FLABEL	/* IPv6 Flow Label */
		OXM_OF_ICMPV6_TYPE	/* ICMPv6 type. */
		OXM_OF_ICMPV6_CODE	/* ICMPv6 code. */
		OXM_OF_IPV6_ND_TARGET	/* Target address for ND. */
		OXM_OF_IPV6_ND_SLL	/* Source link-layer for ND. */
		OXM_OF_IPV6_ND_TLL	/* Target link-layer for ND. */
		OXM_OF_MPLS_LABEL	/* MPLS label. */
		OXM_OF_MPLS_TC	/* MPLS TC. */
7.2.2.8		Experimenter Flow Match Fields	Experimenter-specific ow match fields,may be defined u the oxm_class=OFPXMC_EXPERIMI
7.2.3		Flow Instruction Structures	Flow instructions associated with ow table entry are executed when ow matches the entry
7.2.4		Action Structures	
		OFPAT_OUTPUT = 0,	/* Output to switch port. */
		OFPAT_COPY_TTL_OUT = 11,	/* Copy TTL "outwards" -- from next-to-outermost to outermost */
		OFPAT_COPY_TTL_IN = 12,	/* Copy TTL "inwards" -- from oute to next-to-outermost */
		OFPAT_SET_MPLS_TTL = 15,	/* MPLS TTL */

		OFPAT_DEC_MPLS_TTL = 16,	/* Decrement MPLS TTL */
		OFPAT_PUSH_VLAN = 17,	/* Push a new VLAN tag */
		OFPAT_POP_VLAN = 18,	/* Pop the outer VLAN tag */
		OFPAT_PUSH_MPLS = 19,	/* Push a new MPLS tag */
		OFPAT_POP_MPLS = 20,	/* Pop the outer MPLS tag */
		OFPAT_SET_QUEUE = 21,	/* Set queue id when outputting to */
		OFPAT_GROUP = 22,	/* Apply group. */
		OFPAT_SET_NW_TTL = 23,	/* IP TTL. */
		OFPAT_DEC_NW_TTL = 24,	/* Decrement IP TTL. */
		OFPAT_SET_FIELD = 25,	/* Set a header field using OXM T format. */
		OFPAT_PUSH_PBB = 26	/* Push a new PBB service tag (I- */
		OFPAT_POP_PBB = 27	/* Pop the outer PBB service tag (*/
		OFPAT_EXPERIMENTER = 0xffff	
7.2.5		Experimenter Structure	Experimenter extensions provide standard way for OpenFlow switch offer additional functionality within OpenFlow message type space
7.3	Controller-to-Switch Messages		

7.3.1		Handshake	The OFPT_FEATURES_REQUES message is used by the controller identify the switch and read its ba capabilities
7.3.2		Switch Configuration	The controller is able to set and q configuration parameters in the sv with the OFPT_SET_CONFIG and and OFPT_GET_CONFIG_REQUEST messages, respectively
7.3.3		Flow Table Configuration	Flow entries are modified in the ow table using the OFP_FLOW_M request
7.3.4		Modify State Messages	
7.3.4.1		Modify Flow Table Message	The controller can configure the dynamic state in a ow table with the OFP_TABLE_M request
7.3.4.2		Modify Flow Entry Message	Modifications to a ow table from the controller are do with the OFPT_FLOW_MOD mes
7.3.4.3		Modify Group Entry Message	Modifications to the group table fr controller are done with the OFPT_GROUP_MOD message
7.3.4.4		Port Modification Message	The controller uses the OFPT_PORT_MOD message to r the behavior of the port
7.3.4.5		Meter Modification Messages	Modifications to a meter from the controller are done with the OFPT_METER_MOD message
7.3.5		Multipart Messages	Multipart messages are used to e requests or replies that potentially a large amount of dataand would always fit in a single OpenFlow message, which is limited to 64KE

7.3.5.1		Description	Information about the switch manufacturer, hardware revision, software revision, serial number, a adescription field is available from OFPMP_DESC multipart request
7.3.5.2		Individual Flow Statistics	Information about individual ow entries is requested with the OFPMP_FLOW multipart request
7.3.5.3		Aggregate Flow Statistics	Aggregate information about mult ow entries is requested with the OFPMP_AGGREGATE multipart request type
7.3.5.4		Table Statistics	Information about tables is reques with the OFPMP_TABLE multipar request type
7.3.5.5		Table Description	The OFPMP_TABLE_DESC mult request message provides a way the current configuration of the tal a switch, which is set using the OFPT_TABLE_MOD message.
7.3.5.6		Table Features	The OFPMP_TABLE_FEATURES multipart type allows a controller t query for the capabilities of existir tables, and to optionally ask the s to reconfigure its tables to match supplied configura-tion
		Table Features request and reply	If the OFPMP_TABLE_FEATURE request body is empty the switch return an array of struct ofp_table_features containing the capabilities of the currently config flow tables.
		Table Features properties	A property definition contains the property type, length, and any associated data:

7.3.5.7		Port Statistics	Information about ports statistics i requested with the OFPMP_PORT_STATS multipart request type
7.3.5.8		Port Description	The port description request OFPMP_PORT_DESCRIPTION enables the controller to get a description of allthe ports in the sy that support OpenFlow
7.3.5.9		Queue Statistics	The OFPMP_QUEUE_STATS mu request message provides queue statistics for one or more ports an or more queues
7.3.5.10		Queue Descriptions	The controller can query the switc configuredqueues on a port using OFPMP_QUEUE_DESC multipar request
7.3.5.11		Group Statistics	The OFPMP_GROUP multipart re message provides statistics for or more groups
7.3.5.12		Group Description	The OFPMP_GROUP_DESC mu request message provides a way the set of groups on a switch alon their corresponding bucket action
7.3.5.13		Group Features	The OFPMP_GROUP_FEATURE multipart request message provid way to list the capabilities of grou switch
7.3.5.14		Meter Statistics	The OFPMT_METER stats reque message provides statistics for or more meters.
7.3.5.15		Meter Configuration Statistics	The OFPMT_METER_CONFIG s request message provides configu for one or more meter

7.3.5.16		Meter Features Statistics	The OFPMT_METER_FEATURE request message provides the set features of the metering subsyste
7.3.5.17		Flow monitoring	The OFPMP_FLOW_MONITOR multipart type allows a controller t manage ow monitors, that keep trackof cha to the flow tables
		Flow monitoring request	Flow monitor configuration is done a OFPMP_FLOW_MONITOR mu request
		Flow monitoring reply	When the switch received a OFPMP_FLOW_MONITOR multi request, it replies to it using aOFPMP_FLOW_MONITOR mult reply, the transaction id (xid) of th must be the same as therequest
		Flow monitoring pause	OpenFlow messages for ow monitor notifications can over ow the buffer space available to th switcheither temporarily or more permanently
7.3.5.18		Experimenter Multipart	Experimenter-specific multipart messages are requested with the OFPMP_EXPERIMENTER multip type
7.3.6		Packet-Out Message	When the controller wishes to sen packet out through the datapath, i the OFPT_PACKET_OUTmessag
7.3.7		Barrier Message	When the controller wants to ensu message dependencies have bee or wants to receive notifica-tions f completed operations, it may use OFPT_BARRIER_REQUEST mes

7.3.8		Role Request Message	When the controller wants to char role, it uses the OFPT_ROLE_REQUEST messag
7.3.9		Bundle messages	
7.3.9.1		Bundle control messages	The controller can create, destroy commit bundles with the OFPT_BUNDLE_CONTROL requ
7.3.9.2		Bundle Add message	The controller can add requests to bundle using the OFPT_BUNDLE_ADD_MESSAG message
7.3.9.3		Bundle flags	Bundle flags enable to modify the behavio bundle
7.3.9.4		Bundle properties	A property definition contains the property type, length, and any associated data:
7.3.9.5		Creating and opening a bundle	To create a bundle, the controller a OFPT_BUNDLE_CONTROL me with typeOFPBCT_OPEN_REQU
7.3.9.6		Adding messages to a bundle	The switch adds message to a bu using the OFPT_BUNDLE_ADD_MESSAG
7.3.9.7		Closing a bundle	To finish recording a bundle, the controller may sends a OFPT_BUNDLE_CONTROL mes with typeOFPBCT_CLOSE_REQ
7.3.9.8		Committing Bundles	To finish and apply the bundle, th controller sends a OFPT_BUNDLE_CONTROL mes with typeOFPBCT_COMMIT_REC

7.3.9.9		Discarding Bundles	To finish and discard the bundle, t controller sends a OFPT_BUNDLE_CONTROL mes with typeOFPBCT_DISCARD_REQUE
7.3.9.10		Other bundle error conditions	If a OFPT_BUNDLE_CONTROL message contains an invalid type switch must reject the request and an ofp_error_msg with OFPET_BUNDLE_FAILED type a OFPBFC_BAD_TYPE code.
7.3.10		Set Asynchronous Configuration Message	The switch manages a per-contro asynchronous configuration, whic defines the asynchronousmessag it wants to receive (other than erro messages) on a given OpenFlow channel.
7.4	Asynchronous Messages		
7.4.1		Packet-In Message	When packets are received by the datapath and sent to the controlle use the OFPT_PACKET_IN
7.4.2		Flow Removed Message	If the controller has requested to b notified when ow entries time out or are deleted table,the datapath does this with t OFPT_FLOW_REMOVED messa
7.4.3		Port Status Message	As ports are added, modified, and removed from the datapath, the controller needs to be informed w OFPT_PORT_STATUS message

7.4.4		Controller Role Status Message	When a controller has its role cha by the switch, and not directly cha by that controller using aOFPT_ROLE_REQUEST messa the corresponding controller must informed with a OFPT_ROLE_STATUSmessage
7.4.5		Table Status Message	When the table state changes, the controller needs to be informed w OFPT_TABLE_STATUS message
7.4.6		Request Forward Message	When a controller modifies the sta groups and meters, the request th successfully modifies this state m forwarded to other controller
7.5	Symmetric Messages		
7.5.1		Hello	The OFPT_HELLO message cons an OpenFlow header plus a set of variable size hello elements.
7.5.2		Echo Request	An Echo Request message consis an OpenFlow header plus an arbitrary-length data field.
7.5.3		Echo Reply	An Echo Reply message consists OpenFlow header plus the unmod data field of an echo request mes
7.5.4		Error Message	Error messages are used by the s or the controller to notify the other of the connection of problems
7.5.5		Experimenter Message	
A	Header file openflow.h		
B	Release Notes		

B.14	OpenFlow version 1.4.0		
B.14.1		More extensible wire protocol	
B.14.2		More descriptive reasons for packet-in	
B.14.3		Optical port properties	
B.14.4		Flow-removed reason for meter delete	
B.14.5		Flow monitoring	
B.14.6		Role status events	
B.14.7		Eviction	
B.14.8		Vacancy events	
B.14.9		Bundles	
B.14.10		Synchronised tables	
B.14.11		Group and Meter change notifications	
B.14.12		Error code for bad priority	
B.14.13		Error code for Set-async-config	
B.14.14		PBB UCA header field	
B.14.15		Error code for duplicate instruction	
B.14.16		Error code for multipart timeout	
B.14.17		Change default TCP port to 6653	

PicOS OVS OF-1.3 Support

PicOS OVS supports the following features:

Table 1-1 PicOS OVS Feature List — Spec OF-1.3

OpenFlow V1.3 Section #	Title	Features	Additional Feature
1	Introduction	NA	
2	Switch Components	flow tables	
		group table	
3	Glossary		
4	OpenFlow Ports	See Section 4.3 - 4.5	
4.1	OpenFlow ports	See Section 4.3 - 4.5	
4.2	Standard ports	See Section 4.3 - 4.5	
4.3	physical ports	ingress	OpenFlow packets port, processed by packet ingress port throughout the Ope the OpenFlow port received into the O
		output	The OpenFlow pipe packet on an outpu (see 5.9), which de back to the network
		groups	
		hardware interface	
		virtual slicing of hardware interface	

4.4	logical ports	logical ports are switch defined ports that don't correspond directly to a hardwareinterface of the switch	Logical ports are hi may be defined in t methods
			LAG
			tunnels
			loopback interface
		ingress	
		output	
		groups	
		map to various physical port	
		PACKET_IN reports logical port and its underlying physical port (GRE & LAG)	
4.5	local reserved port	ingress	
		output	
		groups	
		ALL	only as an output p
		CONTROLLER	Represent the cont controller
		TABLE	Represent the start
		IN_PORT	used only as an ou its ingress port
		ANY	Can not be used as output port
		LOCAL	Represent the swit Can be used as an port

			The local port enab with the switch via t than via a separate used to implement connection
		NORMAL	non-OpenFlow pipe port
		FLOOD	flooding using the r used only as an ou
			packet out all stand
			but not to the ingres OFPPS_BLOCKED
5	OpenFlow Tables		
5.1	Pipeline Processing	OpenFlow-only	all packets are proc pipeline
		OpenFlow-hybrid	OpenFlow operatio switching operation
			VLAN tag to decide packet using which
			input port to decide using which pipelin
			allow a packet to g to the normal pipeli FLOOD reserved p

		multiple flow tables	
		sequentially numbered, starting at 0.	
		only go forward and not backward	
		last table of the pipeline can not include the Goto instruction	
		table miss behavior configuration	send packets to the
			drop the packet
			the packet is proces numbered table
			the packet is proces
5.2	Flow Table	flow table entry	match fields
			counters

			instructions
5.3	Matching	packet headers	
		ingress port	
		metadata fields	used to pass inform
		state transition	actions applied in a Apply-Actions are r field
		support ANY	matches all possibl
		support arbitrary bitmasks on specific match fields	
		select highest priority flow entry	
		counters associated with the selected flow entry must be updated	
		CHECK_OVERLAP bit on flow mod messages to avoid overlapping entries	
		multiple matching flow entries with the same highest priority	
		support OFPC_FRAG_REASM flag	IP fragments must pipeline processing
		behavior when a switch receives a corrupted packet	
5.4	Group Table	group identifier	a 32 bit unsigned ir

		group type	to determine group
		counters	updated when pack
		action buckets	an ordered list of ac
5.4.1	Group Types	all	Execute all buckets broadcast
			packet clone is drop packet explicitly ou
			support output actio reserved port
		select	Execute one bucke switch-computed se
		indirect	Execute the one de
		fast failover	Execute the first liv with a live port/grou
	ECMP		Hashing
			Round robin
5.5	Per Table Counters	Reference count (active entries)	32 bits
		Packet Lookups	64 bits
		Packet Matches	64 bits
	Per Flow Counters	Received Packets	64 bits
		Received Bytes	64 bits
		Duration (seconds)	32 bits

		Duration (nanoseconds)	32 bits
	Per Port Counters	Received Packets	64 bits
		Transmitted Packets	64 bits
		Received Bytes	64 bits
		Transmitted Bytes	64 bits
		Receive Drops	64 bits
		Transmit Drops	64 bits
		Receive Errors	64 bits
		Transmit Errors	64 bits
		Receive Frame Alignment Errors	64 bits
		Receive Overrun Errors	64 bits
		Receive CRC Errors	64 bits
		Collisions	64 bits
	Per Queue Counters	Transmit Packets	64 bits
		Transmit Bytes	64 bits
		Transmit Overrun Errors	64 bits
	Per Group Counters	Reference Count (flow entries)	32 bits
		Packet Count	64 bits
		Byte Count	64 bits
	Per Bucket Counters	Packet Count	64 bits

		Byte Count	64 bits
5.6	Instructions	The controller can query the switch about which of the "Optional Instruction" it supports	
		Apply-Actions action(s)	Applies the specific any change to the A
		Clear-Actions	Clears all the actior
		Write-Actions action(s)	Merges the specifie action set
		Write-Metadata metadata / mask	Writes the masked metadata field
		Goto-Table next-table-id	Indicates the next t
		Clear-Actions instruction is executed before the Write-Actions instruction	
		Goto-Table is executed last	
		reject a flow entry if it is unable to execute the instructions & return an unsupported flow error	
5.7	Action Set	action set is associated with each packet	
		This set is empty by default	
		action set is carried between flow tables	

		When the instruction set of a flow entry does not contain a Goto-Table instruction, pipeline processing stops and the actions in the action set of the packet are executed	
		action set contains a maximum of one action of each type	
		The actions in an action set are applied in the order specified below	
		1. copy TTL inwards	
		2. pop	
		3. push	
		4. copy TTL outwards	
		5. decrement TTL	
		6. set	
		7. qos	
		8. group	if a group action is the relevant group by this list
		9. output	if no group action is on the port specifie output action in the
			If both an output ac specified in an actio ignored and the gro
			If no output action a specified in an actio

			The execution of gr supports it; a group group, in which cas traverses all the gro configuration
5.8	Action List	Apply-Actions instruction and the Packet-out message include an action list	The actions of an a order specified by t immediately to the
			The effect of those
			If the action list con of the packet is forv the desired port
			If the list contains a packet in its curren relevant group bucl
5.9	Actions	The controller can also query the switch about which of the "Optional Action" it supports	
		Output	support forwarding switch-defined logic reserved ports
		Set-Queue	The set-queue acti packet and is used Quality-of-Service (
		Drop	
		Group	
		Push-Tag/Pop-Tag	order of header fiel ARP/IP, TCP/UDP/

		Push VLAN header	Push a new VLAN Ethertype is used a Only Ethertype 0x8 used
		Pop VLAN header	Pop the outer-most
		Push MPLS header	Push a new MPLS OnlyEthertype 0x88 used.
		Pop MPLS header	Pop the outer-most from the packet.
		Set-Field	
		Set VLAN ID	
		Strip VLAN ID	
		Change-TTL	modify the values c or MPLS TTL in the
			If it is supported, ap outermost-possible
		Set MPLS TTL	8 bits: New MPLS T MPLS TTL. Only ap existing MPLS shim
		Decrement MPLS TTL	Decrement the MPL packets withan exis
		Set IP TTL	Replace the existin and updatethe IP c and IPv6 packets.

		Decrement IP TTL	Decrement the IPv4 and updatethe IP c and IPv6 packets
		Copy TTL outwards	Copy the TTL from outermost headerw MPLS-to-MPLS, or
		Copy TTL inwards	Copy the TTL from next-to-outermost h IP-to-IP, MPLS-to-M
5.9.1	Default values for fields on push	Field values for all fields specified in Table 6 should be copied from existing outer headers to new outer headers	VLAN ID VLAN ID
		New fields listed in Table 6 without corresponding existing fields should be set to zero	VLAN priority VLAI
		Fields in new headers may be overridden by specifying a "set" action for the appropriate field(s) after the push operation	MPLS label MPLS
			MPLS traffic class
			MPLS TTL MPLS T
6	OpenFlow Channel	encrypted using TLS	
		directly over TCP	
6.1	OpenFlow Protocol Overview	controller-to-switch message type	initiated by the cont manage or inspect
		asynchronous message type	initiated by the swit controller of networ switch state

		symmetric message type	initiated by either th sent without solicita
6.1.1	Controller-to-Switch	Features	request the capabil must respond with
		Configuration	set and query confi switch; switch only controller
		Modify-State	add, delete and mo OpenFlow tables a
		Read-State	used by the control switch
		Packet-out	used by the control specified port on th packets received vi
		Barrier	Barrier request/repl controller to ensure been met or to rece operations
6.1.2	Asynchronous	Switches sendasynchronous messages to controllers to denote a packet arrival, switch state change, or error	
		Packet-in	For all packets that entry, the if table co packets forwarded reserved port, a pa controllers
			If the packet-in eve packets then the pa fraction of the pack used by a controlle to forward the pack

			If the packet is buff the original packet be configured. By d
			or table miss it can configuration
			for packet forwarde configured in the ou
		Flow-Removed	only sent for flow w OFPFF_SEND_FL
			generated as the re requests or the swi one of the flow time
		Port-status	send portstatus me configuration or por
		Error	notify controllers of messages
6.1.3	Symmetric	sent without solicitation, in either direction	
		Hello	exchanged betwee upon connection st
		Echo	sent from either the must return an ech
		Experimenter	a standard way for additional functiona message type spac
6.2	Connection Setup	establish communication with a controller at a user-configurable (but otherwise fixed) IP address, using a user-specified port	Traffic to and from run through the Op switch must identify before checking it a

			each side of the co send an OFPT_HE field set to the high supported by the se
			the recipient may c version to be used number that it sent
			If the negotiated ve recipient, the recipi OFPT_ERROR me OFPET_HELLO_F OFPHFC_COMPA connection.
			optionally an ASCII in data
6.3	Multiple Controllers	establish communication withmultiple controllers	controller fail-over
			controller load bala
			switch virtualisation
			switch must connec configured with, an with all of them con
			the reply or error m command must onl connection associa
			Asynchronous mes multiple controllers, each eligible contro message sent whe connection allows i
		The default role of a controller is OFPCR_ROLE_EQUAL	controller has full a equal to other contr

			controller receives messages (such as
			The controller can s commands to modi
			The switch does nc resource sharing be
		controller can request its role to be changed to OFPCR_ROLE_SLAVE	controller has read-
			controller does not messages, apart fr
			The controller is de controller-to-switch state of the switch, OFPT_FLOW_MOI OFPT_PORT_MOI
			If the controller sen the switch must rep message with a typ OFPET_BAD_REQ OFPBRC_IS_SLAV
			Othercontroller-to-s OFPT_STATS_RE OFPT_ROLE_REQ normally
		A controller can request its role to be changed to OFPCR_ROLE_MASTER	the switch makes s controller in this rol
			When a controller c OFPCR_ROLE_MA other controllers wh OFPCR_ROLE_MA OFPCR_ROLE_SL

			When the switch pe message is genera is changed (in mos longer reachable)
		A switch may be simultaneously connected to multiple controllers in Equal state, multiple controllers in Slave state, and at most one controller in Master state	Each controller may switch via a OFPT_ and the switch mus controller connectio role at any time
			To detect out-of-orc master/slave transi OFPT_ROLE_REQ 64-bit sequence nu identifies a given m
		On receiving a OFPT_ROLE_REQUEST with role equal to OFPCR_ROLE_MASTER or OFPCR_ROLE_SLAVEthe switch must compare the generation_id in the message against the largest generation id seen sofar	A message with a g previously seen ge stale and discarded stale messages wit typeOFPET_ROLE code OFPRRFC_S
6.4	Connection Interruption	a switch loses contact with all controllers, the switch should immediately enter either "fail secure mode" or "fail standalone mode"	In "fail secure mode behavior is that pac to the controllers ar
			Flows should contir timeouts in "fail sec
			In "fail standalone r packets using the C words, the switch a switch or router
			Upon connecting to flow entries remain
			The controller then flow entries, if desir

		The first time a switch starts up, it will operate in either "fail secure mode" or "fail standalone mode" mode, until is successfully connects to a controller	
6.5	Encryption	The switch and controller may communicate through a TLS connection	The TLS connectio startup to the contr default on TCP por
			Each switch must b certificate for authe (controller certificat authenticating to th
6.6	Message Handling	Message Delivery	Messages are guar connection fails ent
		Message Processing	Switches must proc from a controller in reply
			If a switch cannot c received from a cor error message
			switches must send asynchronous mes state changes, such messages
	Message Ordering	Ordering can be ensured through the use of barrier messages	In the absence of b may arbitrarily reor performance
			Messages must no message and the b processed only whe been processed
			messages before a processed before th any resulting replie

			the barrier must the reply sent
			messages after the processing
6.7	Flow Table Modification Messages	OFPFC_ADD	For add requests (O OFPFF_CHECK_O must first check for the requested table
			If an overlap conflic flow entry and the a refuse the addition ofp_error_msg with OFPET_FLOW_MO OFPFMFC_OVERL
			If a flow entry with i priority already resi then that entry, incl cleared from the tal added
			If the OFPFF_RES flow entry counters they should be cop
			No flow-removed m flow entry eliminate the controller wants should explicitly ser old flow prior to add
			If a switch cannot fi table in which to ad switch should send OFPET_FLOW_MO OFPFMFC_TABLE

		OFPFC_MODIFY or OFPFC_MODIFY_STRICT	if a matching entry instructions field of value from the requ idle_timeout, hard_ duration fields are l
			If the OFPFF_RES flow entry counters
			if no flow currently matches the reques flow table modificat
			In the strict version match fields, includ priority, are strictly only an identical flo
			If the match in a flo bitmask for another support, the switch with OFPET_BAD_ OFPBMC_BAD_MA
			If the match in a flo cannot be matched greater than 4095 a values, or a DSCP higher bits set, the ofp_error_msg with and OFPBMC_BAD
			If the match in a flo field that is unsupp must return an ofp_ OFPET_BAD_MAT OFPBMC_BAD_FI
			If the match in a flo field more than onc ofp_error_msg with and OFPBMC_DUP

			If the match in a flo field but fail to spec for example specifi matching the Ether return an ofp_error OFPET_BAD_MAT OFPBMC_BAD_PF
			If the match in a flo bitmask for either th addresses which th switch must return OFPET_BAD_MAT OFPBMC_BAD_DL OFPBMC_BAD_NV
			If an action in a flov group that is not cu or is a reserved grc switch must return OFPET_BAD_ACT OFPBAC_BAD_OU
			If an action in a flov that is invalid, for e with value greater t with an invalid Ethe an ofp_error_msg v type and OFPBAC_
			If an action in a flov operation which is i example, a pop VL specifying no VLAN with a match wildca may optionally reje return an ofp_error OFPET_BAD_ACT OFPBAC_MATCH_

			If any other errors c the flow mod messa ofp_error_msg with OFPET_FLOW_MC OFPFMC_UNKNO
		OFPFC_DELETE or OFPFC_DELETE_STRICT	if a matching entry deleted
			if the entry has the flag set, it should g message
			if no flow currently matches the reques flow table modificat
			In the strict version match fields, includ priority, are strictly only an identical flo
			For non-strict modif flows that match th modified or remove
			In the non-strict ver a flow entry exactly than the description the flow mod the m wildcarded, field ma mod fields such as
			Delete commands destination group o
			If the out_port field OFPP_ANY, it intro matching
			Modify and delete c by cookie value

			Delete commands for table-id to indica be deleted from all
			If the flow modificat invalid table-id, the ofp_error_msg with OFPET_FLOW_MC OFPFMFC_BAD_T
			If the instructions re message are unkno ofp_error_msg with type and OFPBIC_
			If the instructions re message are unsup an ofp_error_msg v OFPET_BAD_INST OFPBIC_UNSUP_I
			If the instructions re and the next-table-i switch must return OFPET_BAD_INST OFPBIC_BAD_TAE
			If the instructions re Write-Metadata and metadata mask val switch must return OFPET_BAD_INST OFPBIC_UNSUP_I OFPBIC_UNSUP_I
			If the bitmasks spe network addresses OFPBMC_BAD_DL used
			If any action referer valid on a switch, th ofp_error_msg with and OFPBAC_BAD

			If the referenced pc e.g. when a linecar or a port is dynamic switch, the switch n sent to the referenc an OFPBAC_BAD_ the flow mod
			If an action list cont the switch can not s the switch should re OFPET_BAD_ACT OFPBAC_UNSUPF
6.8	Flow Removal	switch flow expiry mechanism	is run by the switch and is based on the flow entries
			A non-zero hard_tir entry to be removec seconds, regardles matched
			A non-zero idle_tim entry to be removec packets in the given
			The switch must im remove flow entries of their timeout is e
			When a flow entry i check the flow entry OFPFF_SEND_FL set, the switch mus message to the cor
			Each flow removed description of the fl removal (expiry or d at the time of remov time of removal

6.9	Group Table Modification Messages	OFPGC_ADD	Groups may consis
			A group may also i themselves forwarc supports it.
			The action set for e using the same rule (Section 6.7), with a checks
			If an action in one c unsupported, the sv ofp_error_msg with and code correspor
			if a group entry with already resides in t must refuse to add an ofp_error_msg v OFPET_GROUP_M OFPGMFC_GROU
			If a specified group must refuse to add an ofp_error_msg v OFPET_GROUP_M OFPGMFC_INVAL
			If a switch does not with select groups (than 1), it must refu must send an ofp_e OFPET_GROUP_M OFPGMFC_WEIGH
			If a switch cannot a due to lack of spac ofp_error_msg with OFPET_GROUP_M OFPGMFC_OUT_C

			If a switch cannot a due to restrictions (the number of grou add the group entry ofp_error_msg with OFPET_GROUP_M OFPGMFC_OUT_C
			If a switch cannot a because it does no liveliness configura ofp_error_msg with OFPET_GROUP_M OFPGMFC_WATC
		OFPGC_MODIFY	if a group entry with already resides in t including its type ar removed, and the r
			If a group entry with does not already ex the group mod and OFPET_GROUP_M OFPGMFC_UNKN
		OFPGC_DELETE	if no group entry wi currentlyexists in th recorded, and no g
			To delete all groups specify OFPG_ALL
		Groups	Groups may be cha when at least one g or in more complex
			If a switch does not must send an ofp_e OFPET_GROUP_M OFPGMFC_CHAIN
			A switch may suppo created while chain

			if a group mod is se would be created, t group mod and mu OFPET_GROUP_M OFPGMFC_LOOP
			A switch may suppo forwarded to by oth
			If a switch cannot d referenced by anotl delete the group en ofp_error_msg with OFPET_GROUP_M OFPGMFC_CHAIN
A	Appendix A The OpenFlow Protocol		
A.2.1	Port Structures	The port_no field uniquely identifies a port within a switch	Ports are numbered
		The name field is a null-terminated string containing a human-readable name for the interface	
		port administrative settings support the following states	The OFPPC_PORT port has been admi should not be used
			The OFPPC_NO_F received on that po
			The OFPPC_NO_F OpenFlow should n
			The OFPPC_NO_F OpenFlow should n
			The OFPPFL_NO_ packets on that por should never trigge controller

			the port config bits not changed by the
			If the port config bit through another ad switch sends an OF message to notify t
		The state field describes the port internal state that supports the following states	OFPPS_LINK_DOV link is not present
			The OFPPS_BLOC protocol outside of Spanning Tree, is p with OFPP_FLOOD
			OFPPS_LIVE indic Group
			All port state bits ar changed by the cor
			When the port flags sends an OFPT_PO notify the controller
		The curr, advertised, supported, and peer fields indicate link modes (speed and duplexity), link type (copper/fiber) and link features (auto negotiation and pause)	
		The curr_speed and max_speed fields indicate the current and maximum bit rate (raw transmission speed) of the link in kbps	
A.2.2	Queue Structures	QoS (DSCP & Q mapping?)	An OpenFlow switc Quality-of-Service s queuing mechanisr attach to a port and Flows mapped to a according to that qu

A.2.3	Flow Match Structures	An OpenFlow match is composed of a flow match header and a sequence of zero or more flow match fields	The only valid matc OFPMT_OXM, the OFPMT_STANDAF
			The flow match fiel OpenFlow Extensib
		The OpenFlow specification distinguishes two types of OXM match classes	ONF member class
			ONF reserved class
		Flow Match Fields	
		OXM_OF_IN_PORT	/* Switch input port.
		OXM_OF_IN_PHY_PORT	/* Switch physical ir
		OXM_OF_METADATA	/* Metadata passed
		OXM_OF_ETH_DST	/* Ethernet destinat
		OXM_OF_ETH_SRC	/* Ethernet source a
		OXM_OF_ETH_TYPE	/* Ethernet frame ty
		OXM_OF_VLAN_VID	/* VLAN id. */
		OXM_OF_VLAN_PCP	/* VLAN priority. */
		OXM_OF_IP_DSCP	/* IP DSCP (6 bits i
		OXM_OF_IP_ECN	/* IP ECN (2 bits in
		OXM_OF_IP_PROTO	/* IP protocol. */
		OXM_OF_IPV4_SRC	/* IPv4 source addr
		OXM_OF_IPV4_DST	/* IPv4 destination a
		OXM_OF_TCP_SRC	/* TCP source port.

		OXM_OF_TCP_DST	/* TCP destination
		OXM_OF_UDP_SRC	/* UDP source port
		OXM_OF_UDP_DST	/* UDP destination
		OXM_OF_SCTP_SRC	/* SCTP source por
		OXM_OF_SCTP_DST	/* SCTP destinatior
		OXM_OF_ICMPV4_TYPE	/* ICMP type. */
		OXM_OF_ICMPV4_CODE	/* ICMP code. */
		OXM_OF_ARP_OP	/* ARP opcode. */
		OXM_OF_ARP_SPA	/* ARP source IPv4
		OXM_OF_ARP_TPA	/* ARP target IPv4
		OXM_OF_ARP_SHA	/* ARP source hard
		OXM_OF_ARP_THA	/* ARP target hardw
		OXM_OF_IPV6_SRC	/* IPv6 source addr
		OXM_OF_IPV6_DST	/* IPv6 destination
		OXM_OF_IPV6_FLABEL	/* IPv6 Flow Label
		OXM_OF_ICMPV6_TYPE	/* ICMPv6 type. */
		OXM_OF_ICMPV6_CODE	/* ICMPv6 code. */
		OXM_OF_IPV6_ND_TARGET	/* Target address fo
		OXM_OF_IPV6_ND_SLL	/* Source link-layer
		OXM_OF_IPV6_ND_TLL	/* Target link-layer
		OXM_OF_MPLS_LABEL	/* MPLS label. */

		OXM_OF_MPLS_TC	/* MPLS TC. */
		Required match fields	
		OXM_OF_IN_PORT	Ingress port. This n switch-defined logic
		OXM_OF_ETH_DST	Ethernet source ad bitmask
		OXM_OF_ETH_SRC	Ethernet destinatio bitmask
		OXM_OF_ETH_TYPE	Ethernet type of the after VLAN tags.
		OXM_OF_IP_PROTO	IPv4 or IPv6 protoc
		OXM_OF_IPV4_SRC	IPv4 source addres arbitrary bitmask
		OXM_OF_IPV4_DST	IPv4 destination ad or arbitrary bitmask
		OXM_OF_IPV6_SRC	IPv6 source addres arbitrary bitmask
		OXM_OF_IPV6_DST	IPv6 destination ad or arbitrary bitmask
		OXM_OF_TCP_SRC	TCP source port
		OXM_OF_TCP_DST	TCP destination po
		OXM_OF_UDP_SRC	UDP source port
		OXM_OF_UDP_DST	UDP destination pc
A.2.4	Flow Instruction Structures	See Section 5.6	

A.2.5	Action Structures	A number of actions may be associated with flows, groups or packets. The currently defined action types are	
		OFPAT_OUTPUT = 0,	/* Output to switch
		OFPAT_COPY_TTL_OUT = 11,	/* Copy TTL "outwa to outermost */
		OFPAT_COPY_TTL_IN = 12,	/* Copy TTL "inwar next-to-outermost *
		OFPAT_SET_MPLS_TTL = 15,	/* MPLS TTL */
		OFPAT_DEC_MPLS_TTL = 16,	/* Decrement MPLS
		OFPAT_PUSH_VLAN = 17,	/* Push a new VLA
		OFPAT_POP_VLAN = 18,	/* Pop the outer VL
		OFPAT_PUSH_MPLS = 19,	/* Push a new MPL
		OFPAT_POP_MPLS = 20,	/* Pop the outer MP
		OFPAT_SET_QUEUE = 21,	/* Set queue id whe
		OFPAT_GROUP = 22,	/* Apply group. */
		OFPAT_SET_NW_TTL = 23,	/* IP TTL. */
		OFPAT_DEC_NW_TTL = 24,	/* Decrement IP TT
		OFPAT_SET_FIELD = 25,	/* Set a header field
		OFPAT_EXPERIMENTER = 0xffff	
		The type of a set-field action can be any valid OXM header type	OXM types OFPXM OFPXMT_OFB_ME because those are
		OXM_OF_IN_PHY_PORT	/* Switch physical i

		OXM_OF_ETH_DST	/* Ethernet destinat
		OXM_OF_ETH_SRC	/* Ethernet source a
		OXM_OF_ETH_TYPE	/* Ethernet frame ty
		OXM_OF_VLAN_VID	/* VLAN id. */
		OXM_OF_VLAN_PCP	/* VLAN priority. */
		OXM_OF_IP_DSCP	/* IP DSCP (6 bits i
		OXM_OF_IP_ECN	/* IP ECN (2 bits in
		OXM_OF_IP_PROTO	/* IP protocol. */
		OXM_OF_IPV4_SRC	/* IPv4 source addr
		OXM_OF_IPV4_DST	/* IPv4 destination a
		OXM_OF_TCP_SRC	/* TCP source port.
		OXM_OF_TCP_DST	/* TCP destination p
		OXM_OF_UDP_SRC	/* UDP source port.
		OXM_OF_UDP_DST	/* UDP destination
		OXM_OF_SCTP_SRC	/* SCTP source por
		OXM_OF_SCTP_DST	/* SCTP destinatior
		OXM_OF_ICMPV4_TYPE	/* ICMP type. */
		OXM_OF_ICMPV4_CODE	/* ICMP code. */
		OXM_OF_ARP_OP	/* ARP opcode. */
		OXM_OF_ARP_SPA	/* ARP source IPv4
		OXM_OF_ARP_TPA	/* ARP target IPv4 a
		OXM_OF_ARP_SHA	/* ARP source hard

		OXM_OF_ARP_THA	/* ARP target hardv
		OXM_OF_IPV6_SRC	/* IPv6 source addr
		OXM_OF_IPV6_DST	/* IPv6 destination
		OXM_OF_IPV6_FLABEL	/* IPv6 Flow Label
		OXM_OF_ICMPV6_TYPE	/* ICMPv6 type. */
		OXM_OF_ICMPV6_CODE	/* ICMPv6 code. */
		OXM_OF_IPV6_ND_TARGET	/* Target address fo
		OXM_OF_IPV6_ND_SLL	/* Source link-layer
		OXM_OF_IPV6_ND_TLL	/* Target link-layer
		OXM_OF_MPLS_LABEL	/* MPLS label. */
		OXM_OF_MPLS_TC	/* MPLS TC. */
A.3	Controller-to-Switch Messages		
A.3.1	Handshake	datapath_id	The datapath_id fie datapath. The lowe switch MAC addres to the implementer.
			Use datapath_id to switch instances or
		Capabilities supported by the datapath	OFPC_FLOW_STA */
			OFPC_TABLE_ST statistics. */
			OFPC_PORT_STA */

			OFPC_GROUP_ST statistics. */
			OFPC_IP_REASM IP fragments. */
			OFPC_QUEUE_ST statistics. */
			OFPC_PORT_BLC block looping ports.
A.3.2	Switch Configuration	The controller is able to set and query configuration parameters in the switch with the OFPT_SET_CONFIG and OFPT_GET_CONFIG_REQUEST messages, respectively	
		OFPC_* flags	/* Handling of IP fra **/OFPC_FRAG_NO** handling for fragme
			OFPC_FRAG_DRO */
			OFPC_FRAG_REA (only if OFPC_IP_F
			OFPC_FRAG_MAS
			/* TTL processing - packets **/OFPC_INVALID_T** **<< 2, /** Send packe controller */
		miss_send_len	defines the number to the controller as when configured to
			If this field equals 0 bytes of the packet

			If the value is set to complete packet m message, and shou
A.3.3	Flow Table Configuration	Flow tables are numbered from 0 and can take any number until OFPTT_MAX	OFPTT_MAX = 0xf
		The controller can configure and query table state in the switch with the OFP_TABLE_MOD and OFPST_TABLE_STATS requests, respectively	The switch respond aOFPT_STATS_RI
		OFP_TABLE_MOD	If the table_id is OF applied to all tables
		The config field is a bitmap that is used to configure the default behavior of unmatched packets	OFPTC_TABLE_M Send to controller.
			OFPTC_TABLE_M Continue to the nex (OpenFlow 1.0 beh
			OFPTC_TABLE_M the packet. */
			OFPTC_TABLE_M
A.3.4	Modify State Messages	Modifications to a flow table from the controller are done with the OFPT_FLOW_MOD message	
		Modifications to the group table from the controller are done with the OFPT_GROUP_MOD message	
		The controller uses the OFPT_PORT_MOD message to modify the behavior of the port	

A.3.5	Read State Messages	While the system is running, the datapath may be queried about its current state using the OFPT_STATS_REQUEST message	/* Description of thi request body is em ofp_desc_stats. **/O** reply to OFPST_DE NULL- terminated A **ofp_desc_stats {c mfr_desc[DESC_S** description. **/char h** Hardware descripti **sw_desc[DESC_S** description. **/char serial_num[SERIA** number. **/char dp_** Human readable de
			/* Individual flow sta struct ofp_flow_stat an array of struct of **= 1, /** Body of reply **/struct ofp_flow_s** Length of this entry table flow came fro **duration_sec; /** Tir seconds. **/uint32_t** has been alive in na beyondduration_se of the entry. **/uint16** seconds idle before **hard_timeout; /** Nu expiration. **/uint8_t /uint64_t cookie; /** identifier. **/uint64_t** packets in flow. **/ui** of bytes in flow. **/st** Description of fields **ofp_instruction in** */};

			/* Aggregate flow s struct ofp_aggregat body is struct ofp_a **/OFPST_AGGREG** OFPST_AGGREGA **ofp_aggregate_sta** **packet_count; /** Nu **/uint64_t byte_cou** **/uint32_t flow_cou** **pad[4]; /** Align to 6

			/* Flow table statist empty. The reply b ofp_table_stats. **/O** reply to OFPST_TA **ofp_table_stats {u** table. Lower numbe **/uint8_t pad[7]; /** A **name[OFP_MAX_** **match; /** Bitmap of **indicate the fields** ***/uint64_t wildcard** OFPXMT_**) wildca** **the table. */uint32_** OFPAT_* that are s OFPIT_WRITE_AC **apply_actions; /** B supported by the ta OFPIT_APPLY_AC **write_setfields;/** B **header fields that** **OFPIT_WRITE_AC** **apply_setfields;/** E **header fields that** **OFPIT_APPLY_AC** **metadata_match;** match. **/uint64_t m** metadata table can **/** Bitmap of OFPIT_ **config; /** Bitmap of **max_entries; /** Ma **/uint32_t active_c** entries. **/uint64_t lo** packets looked up i **matched_count; /** table. */};

			/* Port statistics. Th ofp_port_stats_requ array of struct ofp_ **/** Body of reply to O counter is unsuppo **/struct ofp_port_s port_no;uint8_t pa /uint64_t rx_packe** packets. **/uint64_t** transmitted packets Number of received Number of transmit **rx_dropped; /** Num RX. **/uint64_t tx_d** dropped by TX. **/ui** receive errors. This receive errors and s equal to the sum of ***tx_errors; /*** *Numbe super-set of more s should be greater t tx*_**err values (non */uint64_t rx_fram** alignment errors. **/u** Number of packets **rx_crc_err; /** Numb **collisions; /** Numb
			/* Queue statistics f struct ofp_queue_s an array of struct of **/OFPST_QUEUE = consists of an arra structure:struct of port_no;uint32_t q /uint64_t tx_bytes** bytes. **/uint64_t tx_** transmitted packets Number of packets

			/* Group features. T The reply body is s ofp_group_features **/OFPST_GROUP_** reply to OFPST_GF Group features. **/st** **ofp_group_featur** Bitmap of OFPGT_ **capabilities; /** Bitm supported. **/uint32_** number of groups f **actions[4]; /** Bitma supported. */};
			/* Experimenter ext bodies begin with* s ofp_experimenter_s reply bodies are oth **/OFPST_EXPERIM** ofp_stats_request/r OFPST_EXPERIM **ofp_experimenter_** **experimenter; /** Ex same form as in str **/uint32_t exp_type** Experimenter-defin
A.3.6	Queue Configuration Messages	Queue configuration takes place outside the OpenFlow protocol, either through a command line tool	CLI support
		The switch replies back with an ofp_queue_get_config_reply command, containing a list of configured queues	/* Queue configurat **ofp_queue_get_co** **ofp_header heade** **pad[4];struct ofp_** List of configured q
A.3.7	Packet-Out Message	When the controller wishes to send a packet out through the datapath, it uses the OFPT_PACKET_OUT message	

A.3.8	Barrier Message	When the controller wants to ensure message dependencies have been met or wants to receive notifications for completed operations, it may use an OFPT_BARRIER_REQUEST message	Upon receipt, the s previously-received corresponding reply executing any mess RequestRequest. W complete, the switc OFPT_BARRIER_R of the original requ
A.3.9	Role Request Message	When the controller wants to change its role, it uses the OFPT_ROLE_REQUEST message and can have the following values	OFPCR_ROLE_NO current role. */
			OFPCR_ROLE_EQ access. */
			OFPCR_ROLE_MA most one master. */
			OFPCR_ROLE_SL *
A.4	Asynchronous Messages		
A.4.1	Packet-In Message	Switches that implement buffering are expected to expose, through documentation, both the amount of available buffering, and the length of time before buffers may be reused	A switch should pre reused until it has b or some amount of documentation) has
A.4.2	Flow Removed Message	If the controller has requested to be notified when flows time out or are deleted from tables, the datapath does this with the OFPT_FLOW_REMOVED message	The reason field is following:OFPRR_I idle time exceeded **/OFPRR_HARD_TI** hard_timeout. **/OFP** a DELETE flow mo **= 3, /** Group was re

A.4.3	Port Status Message	As ports are added, modified, and removed from the datapath, the controller needs to be informed with the OFPT_PORT_STATUS message	The status can be c values:OFPPR_AD **/OFPPR_DELETE /OFPPR_MODIFY** port has changed. *
A.4.4	Error Message	There are times that the switch needs to notify the controller of a problem. This is done with the OFPT_ERROR_MSG message	Currently defined e are:OFPET_HELL protocol failed. **/OF** Request was not u **/OFPET_BAD_AC** description. **/OFPE** Error in instruction **4, /** Error in match. **/OFPET_FLOW_M** modifying flow entr **/OFPET_GROUP_** modifying group en **/OFPET_PORT_M** request failed. **/OFI 8, /** Table mod requ **/OFPET_QUEUE_** operation failed. **/OFPET_SWITCH_** Switch config reque **/OFPET_ROLE_RI** Controller Role req **/OFPET_EXPERIM** Experimenter error
A.5	Symmetric Messages	See Section 6.1.3	
B	Appendix B Release Notes		
B.6.6	Vendor Extensions	Vendors are now able to add their own extensions, while still being OpenFlow compliant. The primary way to do this is with the new OFPT_VENDOR message type	

B.6.8	802.1D Spanning Tree	A switch that implements STP must set the new OFPC_STP bit in the 'capabilities' field of its OFPT_FEATURES_REPLY message.	A switch that imple available on all of it not implement it on OFPP_LOCAL)
			The complete set o are:OFPPC_PORT administratively do **1, /** Disable 802.1D **/OFPPC_NO_REC** packets received o **/OFPPC_NO_REC** received 802.1D S **/OFPPC_NO_FLO** this port when flood **<< 5, /** Drop packet **/OFPPC_NO_PAC** packet-in msgs for
		packets received on ports that are disabled by spanning tree must follow the normal flow table processing path	
B.6.21	Behavior Defined When Controller Connection Lost	In the case that the switch loses contact with the controller, the default behavior must be to do nothing - to let flows timeout naturally. Other behaviors can be implemented via vendor-specific command line interface or vendor extension OpenFlow messages	Default behavior su
B.7.1	Failover	switch can be configured with a list of controllers. If the first controller fails, it will automatically switch over to the second controller on the list	

B.7.2	Emergency Flow Cache	The protocol and reference implementation have been extended to allow insertion and management of emergency flow entries. Emergency-specific flow entries are inactive until a switch loses connectivity from the controller	the switch invalidat and copies all eme flow table. Upon co all entries in the flo controller then has cache if needed
B.7.9	Rewrite DSCP in IP ToS header	added Flow action to rewrite the DiffServ CodePoint bits part of the IP ToS field in the IP header.	This enables basic OpenFlow in in son
B.8.1	Slicing	OpenFlow now supports multiple queues per output port. Queues support the ability to provide minimum bandwidth guarantees	the bandwidth alloc configurable
B.9	OpenFlow version 1.1		
B.9.1	Multiple Tables	The switch now expose a pipeline with multiple tables.	
		Flow entry have instruction to control pipeline processing	
		Controller can choose packet traversal of tables via goto instruction	
		Metadata field (64 bits) can be set and match in tables	
		Packet actions can be merged in packet action set	
		Packet action set is executed at the end of pipeline	
		Packet actions can be applied between table stages	

		Table miss can send to controller, continue to next table or drop	To controller only
		Rudimentary table capability and configuration	
B.9.2	Groups	Group indirection to represent a set of ports	
		Group table with 4 types of groups :	All - used for multic
			Select - used for m
			Indirect - simple inc
			Fast Failover - use
		Group action to direct a flow to a group	
B.9.3	Tags : MPLS & VLAN	Support for VLAN and QinQ, adding, modifying and removing VLAN headers	
		Support for MPLS, adding, modifying and removing MPLS shim headers	
B.9.4	Virtual ports	Make port number 32 bits, enable larger number of ports	GRE & LAG
		Enable switch to provide virtual port as OpenFlow ports	
		Augment packet-in to report both virtual and physical ports	
B.9.5	Other changes	Remove 802.1d-specific text from spec	
		Remove Emergency Flow Cache from spec	

		Cookie Enhancements Proposal	
		Set queue action (unbundled from output port)	
		Maskable DL and NW address match fields	
		Add TTL decrement, set and copy actions for IPv4 and MPLS	
		SCTP header matching and rewriting support	
		Set ECN action	
		Connection interruption trigger fail secure or fail standalone mode	
		Define message handling : no loss, may reorder if no barrier	
		Rename VENDOR APIs to EXPERIMENTER APIs	
B.10	OpenFlow version 1.2		
B.10.1	Extensible match support	The Extensible set_field action reuses the OXM encoding defined for matches, and enables to rewrite any header field in a single action (EXT-13)	Deprecate most he
			Introduce generic s
			Reuse match TLV s action
B.10.2	Extensible 'set field' packet rewriting support	Rather than introduce a hard coded field in the packet-in message, the flexible OXM encoding is used to carry packet context	Reuse match TLV s metadata in packet

			Include the 'metada
			Move ingress port a field to OXM encod
			Allow to optionally i TLV structure
B.10.3	Extensible context expression in 'packet-in'	Rather than introduce a hard coded field in the packet-in message, the flexible OXM encoding is used to carry packet context	Reuse match TLV s metadata in packet
			Include the 'metada
			Move ingress port a field to OXM encod
			Allow to optionally i TLV structure
B.10.4	Extensible Error messages via experimenter error type	An experimenter error code has been added, enabling experimenter functionality to generate custom error messages (EXT-2). The format is identical to other experimenter APIs	
B.10.5	IPv6 support added	Basic support for IPv6 match and header rewrite has been added	Added support for r address, destinatio traffic class, ICMPv neighbor discovery
			Added support for r (EXT-36)

B.10.6	Simplified behaviour of flow-mod request	The behaviour of flow-mod request has been simplified (EXT-30)	MODIFY and MOD insert new flows in
			New flag OFPFF R counter reset
			Remove quirky beh
B.10.7	Removed packet parsing specification	The match fields are only defined logically	OpenFlow does no packets
			Parsing consistenc pre-requisite
B.10.8	Controller role change mechanism	The controller role change mechanism is a simple mechanism to support multiple controllers for failover (EXT-39)	the switch only nee controller to help th mechanism
			Simple mechanism for failover
			Switches may now in parallel
			Enable each contro equal, master or sla
B.10.9	Other changes	Per-table metadata bitmask capabilities (EXT-34)	
		Rudimentary group capabilities (EXT-61)	
		Add hard timeout info in flow-removed messages (OFP-283)	
		Add ability for controller to detect STP support(OFP-285)	

		Turn off packet buffering with OFPCML NO BUFFER (EXT-45)	
		Added ability to query all queues (EXT-15)	
		Added experimenter queue property (EXT-16)	
		Added max-rate queue property (EXT-21)	
		Enable deleting flow in all tables (EXT-10)	
		Enable switch to check chaining when deleting groups (EXT-12)	
		Enable controller to disable buffering (EXT-45)	
		Virtual ports renamed logical ports (EXT-78)	
		New error messages (EXT-1, EXT-2, EXT-12, EXT-13, EXT-39, EXT-74 and EXT-82)	
		Include release notes into the specification document	
		Many other bug fixes, rewording and clarifications	
	OpenFlow 1.3		
	Per flow meters	Flexible meter framework based on per-flow meters and meter bands.	
		ˆ Meter statistics, including per band statistics.	1 band per meter
		ˆ Enable to attach meters flexibly to flow entries.	

		ˆ Simple rate-limiter support (drop packets	
	Per connection event filtering	Add asynchronous message filter for each controller connection	
	Per connection event filtering	Add asynchronous message filter for each controller connection	
		Set default lter value to match OpenFlow 1.2 behaviour	
		Remove OFPC_INVALID_TTL_TO_CONTROLLER config flag	
	Auxiliary connections	Auxiliary connections are mostly useful to carry packet-in and packet-out messages	
	MPLS BoS matching	match the Bottom of Stack bit (BoS) from the MPLS header (EXT-85). The BoS bit indicates if other MPLS shim header are in the payload of the present MPLS packet, and matching this bit can help to disambiguate case where the MPLS label is present MPLS packet, and matching this bit can help to disambiguate case where the MPLS label is reused across levels of MPLS encapsulation	
	Provider Backbone Bridging tagging	Push and Pop operation to add PBB header as a tag.ˆ	
		New OXM field to match I-SID for the PBB header	PBB-MPLS-VLAN c
	Rework tag order	the nal order of tags in a packet is dictated by the order of the taggingoperations, each tagging operation adds its tag in the outermost position	Remove defined or specification.
			ˆ Tags are now alw possible position.
			ˆ Action-list can adc

			ˆ Tag order is prede action-set.
	Tunnel-ID metadata	a new OXM field that expose to the OpenFlow pipeline metadata associated with the logical port, most commonly the demultiplexing eld from the encapsulation header	if the logical port pe tunnel-id eld would the GRE header. b the tunnel-id match
	Cookies in packet-in		
	Duration for stats	duration field was added to most statistics, including port statistics, group statistics, queue statistics and meter statistics	
	On demand flow counters	New flow-mod flags have been added to disable packet and byte counters on a per-flow basis	

Basic configuration in OVS mode

Accessing the Switch

Once you have Access to the switch and Configured PicOS in OVS Mode, you need to configure the IP address and default gateway for the equipment.

To configure properly the IP address for the management of the device, you should use the configuration script "picos_boot" as described in PicOS Mode Selection.
An alternative is to use the PicOS configuration file.

For accessing the switch through the front port instead of the management interface, some openflow flows need to be configured in the system to redirect the management traffic to the control plane of the switch. This is only needed if the swtich cannot be managed by the management interface.

Here is an example:

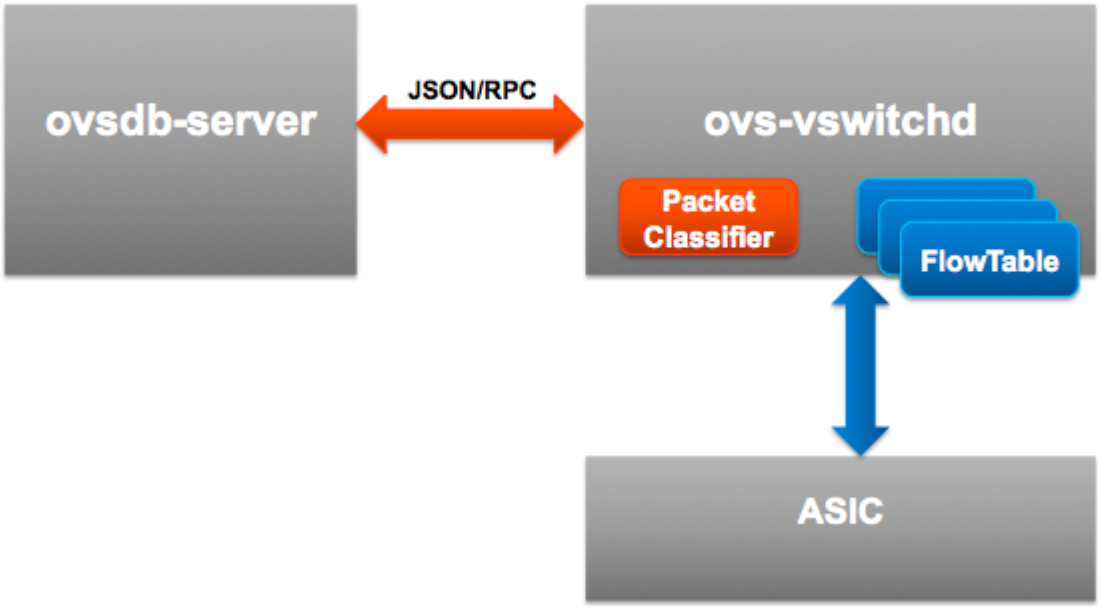
If the bridge br0 has the MAC address c8:0a:a9:04:49:19, in order to access the switch from the inband ports.

```
root@PicOS-OVS#ovs-ofctl add-flow br0
priority=65300,in_port=local,dl_src=c8:0a:a9:04:49:19,actions=all
root@PicOS-OVS#ovs-ofctl add-flow br0
priority=65300,dl_dst=c8:0a:a9:04:49:19,actions=local
root@PicOS-OVS#ovs-ofctl add-flow br0
priority=65300,dl_dst=FF:FF:FF:FF:FF:FF,actions=all,local
```

Understand the OVS Components

OVS as several components:

- ovsdb-server: A database holding switch level information (ports information for example)
- ovs-vswitchd: This is the core component in the system which store the openflow rules and can forward packets.
- openvswitch_mod.ko: This is the Kernel modules doing most of the packet forwarding for OVS. This module is not loaded in PicOS accelerated OVS and is replaced by the ASIC doing the packet forwarding.
- A CLI to control and manipulate those elements.



Understand the OVS CLI

3 Commands are used here to control and monitor the OVS functionalities:

- ovs-vsctl common commands: Used to control the ovsdb-server (create bridges, add ports, configure ports, ...)
- ovs-appctl common commands: Used to control the status of the ovs-vswitchd
- ovs-ofctl common commands: A module to send openflow query. This can be used to manipulate the flows in the ovs-vswitchd

Each of those commands have a "man" page on the unix shell.

```
admin@PicOS-OVS$man ovs-ofctl
ovs-ofctl(8)                  Open vSwitch Manual
ovs-ofctl(8)
NAME
       ovs-ofctl - administer OpenFlow switches
SYNOPSIS
       ovs-ofctl [options] command [switch] [args...]
DESCRIPTION
       The  ovs-ofctl program is a command line tool for monitoring and
admin-
       istering OpenFlow switches.  It can also show the current state  of
an
       OpenFlow  switch, including features, configuration, and table
entries.
       It should work with any OpenFlow switch, not just Open vSwitch.
```

Troubleshooting OVS

Verify that the switch is running in OVS Mode

See the PicOS Troubleshooting Page.

when the switch boots in the OVS mode, there should be 2 processes that must have started - ovsdb-server and ovs-vswitchd.

```
admin@XorPlus$ps -ef | grep ovs
root      1356     1  0 Jan26 ?        00:00:10 /ovs/sbin/ovsdb-server
/ovs/ovs-vswitchd.conf.db --pidfile
--remote=punix:/ovs/var/run/openvswitch/db.sock
root      1358     1  0 Jan26 ?        00:19:07 /ovs/sbin/ovs-vswitchd
--enable-shared-lcmgr
```

In case of Crossflow mode, besides the processes listed above, the router stack must have been initialized:

```
admin@XorPlus$ps -ef | grep pica
root     12430     1  0 Jan07 ?        00:05:49 pica_cardmgr
root     12432     1  0 Jan07 ?        01:03:19 pica_sif
root     12439     1  0 Jan07 ?        00:08:45 pica_lacp
root     12441     1 19 Jan07 ?        4-10:50:14 pica_lcmgr
root     12447     1  0 Jan07 ?        00:09:58 pica_login
root     13218     1  0 Jan07 ?        00:20:47 pica_mstp
root     13236     1  0 Jan07 ?        01:25:30 /pica/bin/xorp_rtrmgr -d -L
local0.info -P /var/run/xorp_rtrmgr.pid
```

Check the Bridge and Port configurations

For the bridge and ports to forward flows in hardware it is required that the datapath_type configured for each entity be set to "pica8"

```
admin@PicOS-OVS$ovs-vsctl show
ac9e5b1e-4234-4158-9214-5660b9343779
  Bridge east
      Controller "tcp:172.16.0.142:6653"
          is_connected: true
      fail_mode: standalone
      Port "ae1"
          tag: 1
          Interface "ae1"
              type: "pica8_lag"
              options: {lacp-mode=active, lacp-system-priority="32768",
lacp-time=slow, lag_type=lacp, link_speed=auto, members="qe-1/1/2"}
      Port "te-1/1/2"
          tag: 1
          Interface "te-1/1/2"
              type: "pica8"
              options: {flow_ctl=none, link_speed=auto}
      Port "te-1/1/1"
          tag: 1
          Interface "te-1/1/1"
              type: "pica8"
              options: {flow_ctl=none, link_speed=auto}

admin@PicOS-OVS$ovs-ofctl show east
OFPT_FEATURES_REPLY (OF1.4) (xid=0x2): dpid:1deb0ae61be44040
n_tables:254, n_buffers:256
capabilities: FLOW_STATS TABLE_STATS PORT_STATS GROUP_STATS
OFPST_PORT_DESC reply (OF1.4) (xid=0x4):
 1(te-1/1/1): addr:ff:ff:ff:ff:ff:00
     config:     0
     state:      LINK_UP
     current:    1GB-FD COPPER
     advertised: 1GB-FD 10GB-FD FIBER
     supported:  10MB-FD 100MB-FD 1GB-FD 10GB-FD FIBER AUTO_NEG
     speed: 1000 Mbps now, 10000 Mbps max
 2(te-1/1/2): addr:ff:ff:ff:ff:ff:00
     config:     0
     state:      LINK_DOWN
     current:    1GB-FD COPPER
     advertised: 1GB-FD 10GB-FD FIBER
     supported:  10MB-FD 100MB-FD 1GB-FD 10GB-FD FIBER AUTO_NEG
     speed: 1000 Mbps now, 10000 Mbps max
 1025(ae1): addr:ff:ff:ff:ff:ff:00
     config:     0
     state:      LINK_UP
     current:    1GB-FD COPPER
     advertised: 1GB-FD 10GB-FD FIBER
     supported:  10MB-FD 100MB-FD 1GB-FD 10GB-FD FIBER AUTO_NEG
     speed: 1000 Mbps now, 10000 Mbps max
 LOCAL(east): addr:0a:e6:1b:e4:40:40
     config:     0
     state:      LINK_UP
     current:    10MB-FD COPPER
     supported:  10MB-FD COPPER
     speed: 10 Mbps now, 10 Mbps max
OFPT_GET_CONFIG_REPLY (OF1.4) (xid=0x6): frags=normal miss_send_len=0
admin@PicOS-OVS$
```

Once the ports are configured and verified, flows can be managed on the OVS.

Check flow discrepancy between the control plane and the hardware

To check the flows in the ovs-vswitchd:

```
admin@PicOS-OVS$ovs-ofctl dump-tables br0 | grep -v active=0:
  0: active=4, lookup=n/a, matched=n/a

admin@PicOS-OVS$ovs-ofctl dump-flows br0
OFPST_FLOW reply (OF1.4) (xid=0x2):
 cookie=0x0, duration=1449.903s, table=0, n_packets=n/a, n_bytes=0,
in_port=1,dl_src=00:00:3d:a6:c8:f2 actions=output:2
 cookie=0x0, duration=1444.537s, table=0, n_packets=n/a, n_bytes=0,
in_port=1,dl_src=00:00:3d:a6:c9:14 actions=output:1
 cookie=0x0, duration=71723.842s, table=0, n_packets=n/a, n_bytes=0,
mpls,in_port=1,dl_vlan=1,mpls_label=10 actions=output:3
 cookie=0x0, duration=74839.581s, table=0, n_packets=n/a, n_bytes=923443200,
in_port=1 actions=output:2
```

To show the hardware flows:

```
admin@PicOS-OVS$ovs-appctl pica/dump-flows
#24 normal permanent priority=32769,in_port=1,dl_src=00:00:3d:a6:c8:f2,
actions:2
#23 normal permanent priority=32769,in_port=1,dl_src=00:00:3d:a6:c9:14,
actions:1
#22 normal permanent priority=32769,mpls,in_port=1,dl_vlan=1,mpls_label=10,
actions:3
#21 normal permanent priority=32769,in_port=1, actions:2
#20 normal permanent priority=0, actions:drop
Total 5 flows in HW.
```

Show the full ovsdb database

```
ovsdb-client dump
```

Configuring Open vSwitch

The port ranges in PicOS are as follows

Physical Port	1-1023
LAG Port	1025-2047
Router Port	2049-3071
GRE Port	3073-4095
VXLAN Port	4097-5119
L2GRE Port	5121-6143

Configure ovsdb-server locally:

Check the ovsdb-server state on switch, '–remote=ptcp:6640:10.10.51.138' means ovsdb-server listenning the IP 10.10.51.138 and PORT 6640. '10.10.51.138' is the IP of switch management-ethernet interface.

```
root@PicOS-OVS$ps aux|grep ovs
root      4281  0.0  0.4   6412  2496 ?        S    01:36   0:00 ovsdb-server
/ovs/ovs-vswitchd.conf.db --pidfile --remote=ptcp:6640:10.10.51.138
--remote=punix:/ovs/var/run/openvswitch/db.sock
root      4286  0.0  1.0  33456  5176 ?        S    01:36   0:00 ovs-vswitchd
--pidfile=ovs-vswitchd.pid --overwrite-pidfile
root      4372  0.0  0.1   2128   684 ttyS0    S+   01:37   0:00 grep
--color=auto ovs
```

configure ovs-vswitchd by local,

```
root@PicOS-OVS$ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
root@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/1 vlan_mode=trunk tag=1 -- set
interface qe-1/1/1 type=pica8
root@PicOS-OVS$ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1
type=pica8_vxlan options:remote_ip=10.10.10.2 options:local_ip=10.10.10.1
options:vlan=1 options:vnid=1122867 options:udp_dst_port=4789
options:src_mac=C8:0A:A9:04:49:1A  options:dst_mac=C8:0A:A9:9E:14:A5
options:egress_port=qe-1/1/1
root@PicOS-OVS$ovs-vsctl set-controller br0 tcp:10.10.51.51:6633
```

Configure ovsdb-server remotely:

Check the ovsdb-server state on switch,

```
root@PicOS-OVS$ps aux|grep ovs
root      4281  0.0  0.4   6412  2496 ?        S    01:36   0:00 ovsdb-server
/ovs/ovs-vswitchd.conf.db --pidfile --remote=ptcp:6640:10.10.51.138
--remote=punix:/ovs/var/run/openvswitch/db.sock
root      4286  0.0  1.0  33456  5176 ?        S    01:36   0:00 ovs-vswitchd
--pidfile=ovs-vswitchd.pid --overwrite-pidfile
root      4372  0.0  0.1   2128   684 ttyS0    S+   01:37   0:00 grep
--color=auto ovs
```

Configure ovs-vswitchd by remote server 10.10.50.42,

```
root@dev-42:~# ovs-vsctl --db=tcp:10.10.51.138:6640 add-br br0 -- set bridge
br0 datapath_type=pica8
root@dev-42:~# ovs-vsctl --db=tcp:10.10.51.138:6640 add-port br0 qe-1/1/1
vlan_mode=trunk tag=1 -- set interface qe-1/1/1 type=pica8
root@dev-42:~# ovs-vsctl --db=tcp:10.10.51.138:6640 add-port br0 vxlan1 --
set interface vxlan1 type=pica8_vxlan options:remote_ip=10.10.10.2
options:local_ip=10.10.10.1 options:vlan=1 options:vnid=1122867
options:udp_dst_port=4789 options:src_mac=C8:0A:A9:04:49:1A
options:dst_mac=C8:0A:A9:9E:14:A5  options:egress_port=qe-1/1/1
root@dev-42:~# ovs-vsctl --db=tcp:10.10.51.138:6640 set-controller br0
tcp:10.10.51.51:6633
```

- Creating a Bridge
- Connecting to a Controller
- Setting the Port Link Speed
- Configuring a Trunk Port
- 40G Changes to 4*10G in OVS
    - 40G Changes to 4*10G in OVS mode on P-5101
    - 40G Changes to 4*10G in OVS mode on P-5401
    - 40G Changes to 4*10G in OVS mode on as6701_32x
    - 40G Changes to 4*10G in OVS mode on P-3922
    - 40G Changes to 4*10G in OVS mode on P-3920
    - 40G Changes to 4*10G in OVS mode on as5712_54x
- Configuring sFlow v5
- Configuring NetFlow
- Configuring Port Mirroring
- Configuring IPv4 OpenFlow
- Configure GRE Tunneling
- Configuring MPLS
- Configuring LAG and LACP
- Creating a Group Table

- Configuring ECMP
- Class of Service Mapping for QoS
- Configuring QoS Queue
- Configuring OpenFlow Meter
- Configuring QinQ
- Configuring OpenFlow Provider Backbone Bridge
- Configuring OpenFlow Loopback
- Enabling Loopback Interface
- Optimizing TCAM Usage
- Configuring Layer 2 over GRE on 5101 and 5401 Switches
- Configuring VXLAN
- Configuring Multi-Table
- Configuring Network Address Translation
- ASIC Limitation
- Configuring CFM
- OVS Configuration File

Creating a Bridge

You can create one or more bridges on a PICA8 switch. Note that each physical port can be added to one and only one bridge.

Adding Ports to a Bridge

In the example below, you create a bridge named **br0** using the **set bridge** command. With the **add-port** command. add access ports, **ge-1/1/1** and **ge-1/1/2**, to **br0**. The default VLAN-ID for both ports is 1.

```
root@PicOS-OVS$ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
device br0 entered promiscuous mode
root@PicOS-OVS$
root@PicOS-OVS$ovs-vsctl add-port br0 ge-1/1/1 vlan_mode=access tag=1 -- set
Interface ge-1/1/1 type=pica8
root@PicOS-OVS$ovs-vsctl add-port br0 ge-1/1/2 vlan_mode=access tag=1 -- set
Interface ge-1/1/2 type=pica8
root@PicOS-OVS$
```

If you allow using DAC line,you should enable DAC.

```
root@PicOS-OVS$ovs-vsctl add-port br0 ge-1/1/1 vlan_mode=trunk tag=1 -- set
Interface ge-1/1/1 type=pica8 options:is_dac=true
```

Adding the Default VLAN-ID

In the example below, add the trunk port **ge-1/1/3** to bridge **br0** with the default VLAN-ID is 1000.

```
root@PicOS-OVS$ovs-vsctl add-port br0 ge-1/1/3 vlan_mode=trunk tag=1000
trunks=1000 -- set Interface ge-1/1/3 type=pica8
root@PicOS-OVS$
```

Viewing the Bridge Settings

Use the **show <bridge_name>** command to view the bridge details.

```
root@PicOS-OVS$ovs-ofctl show br0
OFPT_FEATURES_REPLY (xid=0x1): ver:0x1, dpid:0000e89a8f503d30
n_tables:1, n_buffers:256
features: capabilities:0x87, actions:0x3f
1(ge-1/1/1): addr:e8:9a:8f:50:3d:30
config: 0
state: LINK_DOWN
current: 10MB-FD COPPER AUTO_NEG AUTO_PAUSE AUTO_PAUSE_ASYM
advertised: 10MB-FD AUTO_PAUSE
supported: 10MB-HD 10MB-FD 100MB-HD 100MB-FD 1GB-FD AUTO_NEG AUTO_PAUSE
AUTO_PAUSE_ASYM
peer: 10MB-FD AUTO_PAUSE
2(ge-1/1/2): addr:e8:9a:8f:50:3d:30
config: 0
state: LINK_DOWN
current: 10MB-FD COPPER AUTO_NEG AUTO_PAUSE AUTO_PAUSE_ASYM
advertised: 10MB-FD AUTO_PAUSE
supported: 10MB-HD 10MB-FD 100MB-HD 100MB-FD 1GB-FD AUTO_NEG AUTO_PAUSE
AUTO_PAUSE_ASYM
peer: 10MB-FD AUTO_PAUSE
3(ge-1/1/3): addr:e8:9a:8f:50:3d:30
config: 0
state: LINK_DOWN
current: 10MB-FD COPPER AUTO_NEG AUTO_PAUSE AUTO_PAUSE_ASYM
advertised: 10MB-FD AUTO_PAUSE
supported: 10MB-HD 10MB-FD 100MB-HD 100MB-FD 1GB-FD AUTO_NEG AUTO_PAUSE
AUTO_PAUSE_ASYM
peer: 10MB-FD AUTO_PAUSE
LOCAL(br0): addr:e8:9a:8f:50:3d:30
config: PORT_DOWN
state: LINK_DOWN
current: 10MB-FD COPPER
OFPT_GET_CONFIG_REPLY (xid=0x3): frags=normal miss_send_len=0
root@PicOS-OVS$
root@PicOS-OVS$
root@PicOS-OVS$ ovs-vsctl list-ports br0
ge-1/1/1
ge-1/1/2
ge-1/1/3
root@PicOS-OVS$
root@PicOS-OVS$
root@PicOS-OVS$ovs-vsctl list-ifaces br0
ge-1/1/1
ge-1/1/2
ge-1/1/3
root@PicOS-OVS$
root@PicOS-OVS$
```

Deleting the Bridge

To delete the bridge and its ports, use the **del-port** command and then the **del-br <bridge_name>** command.

```
root@PicOS-OVS$ovs-vsctl  del-port br0 ge-1/1/1
root@PicOS-OVS$ovs-vsctl  del-port br0 ge-1/1/2
root@PicOS-OVS$ovs-vsctl  del-port br0 ge-1/1/3
root@PicOS-OVS$ovs-vsctl  del-br br0
```

Connecting to a Controller

Use the OVSDB protocol to connect to a controller. The **ovs-vsctl** command requires an IP address and a port number of the OVS database server. In the example below, the switch connects to an OF controller with an IP address of10.10.53.50 and port number of 6633.

```
root@PicOS-OVS# ovs-vsctl set-controller br0 tcp:10.10.53.50:6633
root@PicOS-OVS#
```

Setting the Port Link Speed

You can set the link speed of each port using the **options:link_speed=1G** as shown in the following example.

```
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/49 vlan_mode=access tag=1 --
set Interface te-1/1/49 type=pica8 options:link_speed=1G
root@PicOS-OVS#
```

Configuring a Trunk Port

PicOS supports 802.1Q trunk ports (since PicOS 2.1). Each port has a default VLAN-ID; by default, the VLAN-ID is 1. If you want a port to belong to more than one VLAN, use the **vlan mode=trunk** command.When you specify one port to a trunk port(tag=1),that means this port is trunk port,it's PVID is the tag number and this port belongs to all the other vlans(2-4094).

In the example below, you specify the VLAN mode to equal *trunk* and then specify the VLANs in the trunks,

```
root@PicOS-OVS# ovs-vsctl add-port br0 ge-1/1/4 vlan_mode=trunk tag=1 -- set
Interface ge-1/1/4 type=pica8
root@PicOS-OVS#
```

The trunk port can carry all VLANs if you do not specify the *trunks* field as shown below.

```
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/1 vlan_mode=trunk tag=1 -- set
Interface te-1/1/1 type=pica8
root@PicOS-OVS#
```

40G Changes to 4*10G in OVS

In OVS mode, P-5401 ports can be configured to one of the following settings. By default, it is in normal mode.

1. normal - 32 x 40G
2. half - 16 x 40G + 64 x 10G
3. max - 8 x 40G + 96 x 10G

In OVS mode, P-5101 ports can be configured to one of the following settings. By default, it is in normal mode.

1. normal : Ports 1-40 work in 10G and ports 41-48 work in 40G.
2. max : Ports 1-40 work in 10G,ports 41-48 work in 4*10G.

In OVS mode, as6701_32x ports can be configured to one of the following settings. By default, it is in normal mode.

1. normal: All 32 ports work in 40G.
2. max: Ports 5-16,21-32 work in 4*10G,ports 1-4,17-20 work in 40G.

In OVS mode, P-3922 ports can be configured to one of the following settings. By default, it is in normal mode.

1. normal: Ports 1-48 work in 10G and ports 48-52 work in 40G.
2. max: Ports 1-48 work in 10G,ports 48-52 work in 4*10G.

In OVS mode, P-3920 ports can be configured to one of the following settings. By default, it is in normal mode.

1. normal: Ports 1-48 work in 10G and ports 48-52 work in 40G.
2. max: Ports 1-48 work in 10G,ports 48-52 work in 4*10G.

In OVS mode,as5712_54x ports can be configured to one of the following settings. By default, it is in normal mode.

1. normal: Ports 1-48 work in 10G and ports 48-54 work in 40G.
2. max: Ports 1-48 work in 10G,ports 48-54 work in 4*10G.

Note:
On P-5101,as6701_32x,P-3922 and P-3920 do not support the mode of half.

Note
If you plug in DAC line on ovs mode,please enable it,the configuration refers to : OVS Configuration Guide/Configuring Open vSwitch/Creating a Bridge/Adding Ports to a Bridge

- 40G Changes to 4*10G in OVS mode on P-5101
- 40G Changes to 4*10G in OVS mode on P-5401
- 40G Changes to 4*10G in OVS mode on as6701_32x
- 40G Changes to 4*10G in OVS mode on P-3922
- 40G Changes to 4*10G in OVS mode on P-3920
- 40G Changes to 4*10G in OVS mode on as5712_54x

40G Changes to 4*10G in OVS mode on P-5101

In OVS mode Configuration

You can set the port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  set-qe-port-mode  [normal | max]
```

After setting pors to different mode, it is mandatory to restart the OVS service in order to make the new state to take effect.

```
admin@PicOS-OVS$sudo  service  picos  restart
```

You can take a look of the current QSFP port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  show-qe-port-mode
```

normal (8 x 40G+40*10G)

When ports are in normal mode, the mapping between physical port and the associated port/interface name is in the following table.

Physical Port number	OVS port/interface name
1	te-1/1/1
2	te-1/1/2
3	te-1/1/3
4	te-1/1/4
5	te-1/1/5

6	te-1/1/6
7	te-1/1/7
8	te-1/1/8
9	te-1/1/9
10	te-1/1/10
11	te-1/1/11
12	te-1/1/12
13	te-1/1/13
14	te-1/1/14
15	te-1/1/15
16	te-1/1/16
17	te-1/1/17
18	te-1/1/18
19	te-1/1/19
20	te-1/1/20
21	te-1/1/21
22	te-1/1/22
23	te-1/1/23
24	te-1/1/24
25	te-1/1/25
26	te-1/1/26
27	te-1/1/27

28	te-1/1/28
29	te-1/1/29
30	te-1/1/30
31	te-1/1/31
32	te-1/1/32
33	te-1/1/33
34	te-1/1/34
35	te-1/1/36
36	te-1/1/37
37	te-1/1/38
39	te-1/1/39
40	te-1/1/40
41	qe -1/1/41
42	qe -1/1/42
43	qe -1/1/43
44	qe -1/1/44
45	qe -1/1/45
46	qe -1/1/46
47	qe -1/1/47
48	qe -1/1/48

max (72x 10G)

When ports are in max mode, the mapping between physical port and the associated port/interface name is in the following table.

Physical Port number	OVS port/interface name
1	te-1/1/1
2	te-1/1/2
3	te-1/1/3
4	te-1/1/4
5	te-1/1/5
6	te-1/1/6
7	te-1/1/7
8	te-1/1/8
9	te-1/1/9
10	te-1/1/10
11	te-1/1/11
12	te-1/1/12
13	te-1/1/13
14	te-1/1/14
15	te-1/1/15
16	te-1/1/16
17	te-1/1/17
18	te-1/1/18

19	te-1/1/19
20	te-1/1/20
21	te-1/1/21
22	te-1/1/22
23	te-1/1/23
24	te-1/1/24
25	te-1/1/25
26	te-1/1/26
27	te-1/1/27
28	te-1/1/28
29	te-1/1/29
30	te-1/1/30
31	te-1/1/31
32	te-1/1/32
33	te-1/1/33
34	te-1/1/34
35	te-1/1/36
36	te-1/1/37
37	te-1/1/38
39	te-1/1/39
40	te-1/1/40
41	4 x 10G

	te -1/1/41
	te -1/1/42
	te -1/1/43
	te -1/1/44
42	4 x 10G
	te -1/1/45
	te -1/1/46
	te -1/1/47
	te -1/1/48
43	4 x 10G
	te -1/1/49
	te -1/1/50
	te -1/1/51
	te -1/1/52
44	4 x 10G
	te -1/1/53
	te -1/1/54
	te -1/1/55
	te -1/1/56
45	4 x 10G
	te -1/1/57
	te -1/1/58

	te -1/1/59
	te -1/1/60
46	4 x 10G
	te -1/1/61
	te -1/1/62
	te -1/1/63
	te -1/1/64
47	4 x 10G
	te -1/1/65
	te -1/1/66
	te -1/1/67
	te -1/1/68
48	4 x 10G
	te -1/1/69
	te -1/1/70
	te -1/1/71
	te -1/1/72

Note

On P-5101 do not support the mode of half.

40G Changes to 4*10G in OVS mode on P-5401

In OVS mode Configuration

You can set the port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  set-qe-port-mode  [normal | half | max]
```

After setting pors to different mode, it is mandatory to restart the OVS service in order to make the new state to take effect.

```
admin@PicOS-OVS$sudo  service  picos  restart
```

You can take a look of the current port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  show-qe-port-mode
```

normal (32 x 40G)

When ports are in normal mode, the mapping between physical port and the associated port/interface name is in the following table.

Physical Port number	OVS port/interface name
1	qe-1/1/1
2	qe-1/1/2
3	qe-1/1/3
4	qe-1/1/4
5	qe-1/1/5
6	qe-1/1/6
7	qe-1/1/7
8	qe-1/1/8
9	qe-1/1/9
10	qe-1/1/10
11	qe-1/1/11
12	qe-1/1/12

13	qe-1/1/13
14	qe-1/1/14
15	qe-1/1/15
16	qe-1/1/16
17	qe-1/1/17
18	qe-1/1/18
19	qe-1/1/19
20	qe-1/1/20
21	qe-1/1/21
22	qe-1/1/22
23	qe-1/1/23
24	qe-1/1/24
25	qe-1/1/25
26	qe-1/1/26
27	qe-1/1/27
28	qe-1/1/28
29	qe-1/1/29
30	qe-1/1/30
31	qe-1/1/31
32	qe-1/1/32

half (16 x 40G + 64 x 10G)

When ports are in half mode, the mapping between physical port and the associated port/interface name is in the following table.

Physical Port number	OVS port/interface name
1	4 x 10G
	te-1/1/1
	te-1/1/2
	te-1/1/3
	te-1/1/4
2	4 x 10G
	te-1/1/5
	te-1/1/6
	te-1/1/7
	te-1/1/8
3	4 x 10G
	te-1/1/9
	te-1/1/10
	te-1/1/11
	te-1/1/12
4	4 x 10G
	te-1/1/13
	te-1/1/14

	te-1/1/15
	te-1/1/16
5	4 x 10G
	te-1/1/17
	te-1/1/18
	te-1/1/19
	te-1/1/20
6	4 x 10G
	te-1/1/21
	te-1/1/22
	te-1/1/23
	te-1/1/24
7	4 x 10G
	te-1/1/25
	te-1/1/26
	te-1/1/27
	te-1/1/28
8	4 x 10G
	te-1/1/29
	te-1/1/30
	te-1/1/31
	te-1/1/32

9	qe-1/1/33
10	qe-1/1/34
11	qe-1/1/35
12	qe-1/1/36
13	qe-1/1/37
14	qe-1/1/38
15	qe-1/1/39
16	qe-1/1/40
17	4 x 10G
	te-1/1/41
	te-1/1/42
	te-1/1/43
	te-1/1/44
18	4 x 10G
	te-1/1/45
	te-1/1/46
	te-1/1/47
	te-1/1/48
19	4 x 10G
	te-1/1/49
	te-1/1/50
	te-1/1/51

	te-1/1/52
20	4 x 10G
	te-1/1/53
	te-1/1/54
	te-1/1/55
	te-1/1/56
21	4 x 10G
	te-1/1/57
	te-1/1/58
	te-1/1/59
	te-1/1/60
22	4 x 10G
	te-1/1/61
	te-1/1/62
	te-1/1/63
	te-1/1/64
23	4 x 10G
	te-1/1/65
	te-1/1/66
	te-1/1/67
	te-1/1/68
24	4 x 10G

	te-1/1/69
	te-1/1/70
	te-1/1/71
	te-1/1/72
25	qe-1/1/73
26	qe-1/1/74
27	qe-1/1/75
28	qe-1/1/76
29	qe-1/1/77
30	qe-1/1/78
31	qe-1/1/79
32	qe-1/1/80

max (8 x 40G + 96 x 10G)

When ports are in max mode, the mapping between physical port and the associated port/interface name is in the following table.

Physical Port number	OVS port/interface name
1	4 x 10G
	te-1/1/1
	te-1/1/2
	te-1/1/3
	te-1/1/4
2	4 x 10G

	te-1/1/5
	te-1/1/6
	te-1/1/7
	te-1/1/8
3	4 x 10G
	te-1/1/9
	te-1/1/10
	te-1/1/11
	te-1/1/12
4	4 x 10G
	te-1/1/13
	te-1/1/14
	te-1/1/15
	te-1/1/16
5	4 x 10G
	te-1/1/17
	te-1/1/18
	te-1/1/19
	te-1/1/20
6	4 x 10G
	te-1/1/21
	te-1/1/22

	te-1/1/23
	te-1/1/24
7	4 x 10G
	te-1/1/25
	te-1/1/26
	te-1/1/27
	te-1/1/28
8	4 x 10G
	te-1/1/29
	te-1/1/30
	te-1/1/31
	te-1/1/32
9	4 x 10G
	te-1/1/33
	te-1/1/34
	te-1/1/35
	te-1/1/36
10	4 x 10G
	te-1/1/37
	te-1/1/38
	te-1/1/39
	te-1/1/40

11	4 x 10G
	te-1/1/41
	te-1/1/42
	te-1/1/43
	te-1/1/44
12	4 x 10G
	te-1/1/45
	te-1/1/46
	te-1/1/47
	te-1/1/48
13	qe-1/1/49
14	qe-1/1/50
15	qe-1/1/51
16	qe-1/1/52
17	4 x 10G
	te-1/1/53
	te-1/1/54
	te-1/1/55
	te-1/1/56
18	4 x 10G
	te-1/1/57
	te-1/1/58

	te-1/1/59
	te-1/1/60
19	4 x 10G
	te-1/1/61
	te-1/1/62
	te-1/1/63
	te-1/1/64
20	4 x 10G
	te-1/1/65
	te-1/1/66
	te-1/1/67
	te-1/1/68
21	4 x 10G
	te-1/1/69
	te-1/1/70
	te-1/1/71
	te-1/1/72
22	4 x 10G
	te-1/1/73
	te-1/1/74
	te-1/1/75
	te-1/1/76

23	4 x 10G
	te-1/1/77
	te-1/1/78
	te-1/1/79
	te-1/1/80
24	4 x 10G
	te-1/1/81
	te-1/1/82
	te-1/1/83
	te-1/1/84
25	4 x 10G
	te-1/1/85
	te-1/1/86
	te-1/1/87
	te-1/1/88
26	4 x 10G
	te-1/1/89
	te-1/1/90
	te-1/1/91
	te-1/1/92
27	4 x 10G
	te-1/1/93

	te-1/1/94
	te-1/1/95
	te-1/1/96
28	4 x 10G
	te-1/1/97
	te-1/1/98
	te-1/1/99
	te-1/1/100
29	qe-1/1/101
30	qe-1/1/102
31	qe-1/1/103
32	qe-1/1/104

40G Changes to 4*10G in OVS mode on as6701_32x

In OVS mode Configuration

You can set the port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  set-qe-port-mode  [normal | max]
```

After setting pors to different mode, it is mandatory to restart the OVS service in order to make the new state to take effect.

```
admin@PicOS-OVS$sudo  service  picos  restart
```

You can take a look of the current port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  show-qe-port-mode
```

normal (32 x 40G)

When ports are in normal mode, the mapping between physical ports , interface names and interface support speed are in the following table.

Physical Port number	OVS port/interface name	interface support speed
1	qe-1/1/1	40Gb/s
2	qe-1/1/2	40Gb/s
3	qe-1/1/3	40Gb/s
4	qe-1/1/4	40Gb/s
5	qe-1/1/5	40Gb/s
6	qe-1/1/6	40Gb/s
7	qe-1/1/7	40Gb/s
8	qe-1/1/8	40Gb/s
9	qe-1/1/9	40Gb/s
10	qe-1/1/10	40Gb/s
11	qe-1/1/11	40Gb/s
12	qe-1/1/12	40Gb/s
13	qe-1/1/13	40Gb/s
14	qe-1/1/14	40Gb/s
15	qe-1/1/15	40Gb/s
16	qe-1/1/16	40Gb/s
17	qe-1/1/17	40Gb/s
18	qe-1/1/18	40Gb/s

19	qe-1/1/19	40Gb/s
20	qe-1/1/20	40Gb/s
21	qe-1/1/21	40Gb/s
22	qe-1/1/22	40Gb/s
23	qe-1/1/23	40Gb/s
24	qe-1/1/24	40Gb/s
25	qe-1/1/25	40Gb/s
26	qe-1/1/26	40Gb/s
27	qe-1/1/27	40Gb/s
28	qe-1/1/28	40Gb/s
29	qe-1/1/29	40Gb/s
30	qe-1/1/30	40Gb/s
31	qe-1/1/31	40Gb/s
32	qe-1/1/32	40Gb/s

max (8 x 40G + 96 x 10G)

When ports are in max mode, the mapping between physical ports and the logical ports/interfaces name are in the following table.

Physical Port number	OVS port/interface name	interface support speed
1	qe-1/1/1	40Gb/s
2	qe-1/1/2	40Gb/s
3	qe-1/1/3	40Gb/s
4	qe-1/1/4	40Gb/s

5	4 x 10G	
	te-1/1/5	10Gb/s
	te-1/1/6	10Gb/s
	te-1/1/7	10Gb/s
	te-1/1/8	10Gb/s
6	4 x 10G	
	te-1/1/9	10Gb/s
	te-1/1/10	10Gb/s
	te-1/1/11	10Gb/s
	te-1/1/12	10Gb/s
7	4 x 10G	
	te-1/1/13	10Gb/s
	te-1/1/14	10Gb/s
	te-1/1/15	10Gb/s
	te-1/1/16	10Gb/s
8	4 x 10G	
	te-1/1/17	10Gb/s
	te-1/1/18	10Gb/s
	te-1/1/19	10Gb/s
	te-1/1/20	10Gb/s
9	4 x 10G	
	te-1/1/21	10Gb/s

	te-1/1/22	10Gb/s
	te-1/1/23	10Gb/s
	te-1/1/24	10Gb/s
10	4 x 10G	
	te-1/1/25	10Gb/s
	te-1/1/26	10Gb/s
	te-1/1/27	10Gb/s
	te-1/1/28	10Gb/s
11	4 x 10G	
	te-1/1/29	10Gb/s
	te-1/1/30	10Gb/s
	te-1/1/31	10Gb/s
	te-1/1/32	10Gb/s
12	4 x 10G	
	te-1/1/33	10Gb/s
	te-1/1/34	10Gb/s
	te-1/1/35	10Gb/s
	te-1/1/36	10Gb/s
13	4 x 10G	
	te-1/1/37	10Gb/s
	te-1/1/38	10Gb/s
	te-1/1/39	10Gb/s

	te-1/1/40	10Gb/s
14	4 x 10G	
	te-1/1/41	10Gb/s
	te-1/1/42	10Gb/s
	te-1/1/43	10Gb/s
	te-1/1/44	10Gb/s
15	4 x 10G	
	te-1/1/45	10Gb/s
	te-1/1/46	10Gb/s
	te-1/1/47	10Gb/s
	te-1/1/48	10Gb/s
16	4 x 10G	
	te-1/1/49	10Gb/s
	te-1/1/50	10Gb/s
	te-1/1/51	10Gb/s
	te-1/1/52	10Gb/s
17	qe-1/1/53	40Gb/s
18	qe-1/1/54	40Gb/s
19	qe-1/1/55	40Gb/s
20	qe-1/1/56	40Gb/s
21	4 x 10G	
	te-1/1/57	10Gb/s

	te-1/1/58	10Gb/s
	te-1/1/59	10Gb/s
	te-1/1/60	10Gb/s
22	4 x 10G	
	te-1/1/61	10Gb/s
	te-1/1/62	10Gb/s
	te-1/1/63	10Gb/s
	te-1/1/64	10Gb/s
23	4 x 10G	
	te-1/1/65	10Gb/s
	te-1/1/66	10Gb/s
	te-1/1/67	10Gb/s
	te-1/1/68	10Gb/s
24	4 x 10G	
	te-1/1/69	10Gb/s
	te-1/1/70	10Gb/s
	te-1/1/71	10Gb/s
	te-1/1/72	10Gb/s
25	4 x 10G	
	te-1/1/73	10Gb/s
	te-1/1/74	10Gb/s
	te-1/1/75	10Gb/s

	te-1/1/76	10Gb/s
26	4 x 10G	
	te-1/1/77	10Gb/s
	te-1/1/78	10Gb/s
	te-1/1/79	10Gb/s
	te-1/1/80	10Gb/s
27	4 x 10G	
	te-1/1/81	10Gb/s
	te-1/1/82	10Gb/s
	te-1/1/83	10Gb/s
	te-1/1/84	10Gb/s
28	4 x 10G	
	te-1/1/85	10Gb/s
	te-1/1/86	10Gb/s
	te-1/1/87	10Gb/s
	te-1/1/88	10Gb/s
29	4 x 10G	
	te-1/1/89	10Gb/s
	te-1/1/90	10Gb/s
	te-1/1/91	10Gb/s
	te-1/1/92	10Gb/s
30	4 x 10G	

	te-1/1/93	10Gb/s
	te-1/1/94	10Gb/s
	te-1/1/95	10Gb/s
	te-1/1/96	10Gb/s
31	4 x 10G	
	te-1/1/97	10Gb/s
	te-1/1/98	10Gb/s
	te-1/1/99	10Gb/s
	te-1/1/100	10Gb/s
32	4 x 10G	
	te-1/1/101	10Gb/s
	te-1/1/102	10Gb/s
	te-1/1/103	10Gb/s
	te-1/1/104	10Gb/s

40G Changes to 4*10G in OVS mode on P-3922

In OVS mode Configuration

You can set the port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  set-qe-port-mode  [normal | max]
```

After setting pors to different mode, it is mandatory to restart the OVS service in order to make the new state to take effect.

```
admin@PicOS-OVS$sudo  service  picos  restart
```

You can take a look of the current port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  show-qe-port-mode
```

normal(48*10G+4*40G)

When ports are in normal mode, the mapping between physical port , interface names and interfaces support speed is in the following table.

Physical Port number	interface name	interface support speed
1	te-1/1/1	10Gb/s and 1Gb/s
2	te-1/1/2	10Gb/s and 1Gb/s
3	te-1/1/3	10Gb/s and 1Gb/s
4	te-1/1/4	10Gb/s and 1Gb/s
5	te-1/1/5	10Gb/s and 1Gb/s
6	te-1/1/6	10Gb/s and 1Gb/s
7	te-1/1/7	10Gb/s and 1Gb/s
8	te-1/1/8	10Gb/s and 1Gb/s
9	te-1/1/9	10Gb/s and 1Gb/s
10	te-1/1/10	10Gb/s and 1Gb/s
11	te-1/1/11	10Gb/s and 1Gb/s
12	te-1/1/12	10Gb/s and 1Gb/s
13	te-1/1/13	10Gb/s and 1Gb/s
14	te-1/1/14	10Gb/s and 1Gb/s
15	te-1/1/15	10Gb/s and 1Gb/s
16	te-1/1/16	10Gb/s and 1Gb/s
17	te-1/1/17	10Gb/s and 1Gb/s

18	te-1/1/18	10Gb/s and 1Gb/s
19	te-1/1/19	10Gb/s and 1Gb/s
20	te-1/1/20	10Gb/s and 1Gb/s
21	te-1/1/21	10Gb/s and 1Gb/s
22	te-1/1/22	10Gb/s and 1Gb/s
23	te-1/1/23	10Gb/s and 1Gb/s
24	te-1/1/24	10Gb/s and 1Gb/s
25	te-1/1/25	10Gb/s and 1Gb/s
26	te-1/1/26	10Gb/s and 1Gb/s
27	te-1/1/27	10Gb/s and 1Gb/s
28	te-1/1/28	10Gb/s and 1Gb/s
29	te-1/1/29	10Gb/s and 1Gb/s
30	te-1/1/30	10Gb/s and 1Gb/s
31	te-1/1/31	10Gb/s and 1Gb/s
32	te-1/1/32	10Gb/s and 1Gb/s
33	te-1/1/33	10Gb/s and 1Gb/s
34	te-1/1/34	10Gb/s and 1Gb/s
35	te-1/1/35	10Gb/s and 1Gb/s
36	te-1/1/36	10Gb/s and 1Gb/s
37	te-1/1/37	10Gb/s and 1Gb/s
38	te-1/1/38	10Gb/s and 1Gb/s
39	te-1/1/39	10Gb/s and 1Gb/s

40	te-1/1/40	10Gb/s and 1Gb/s
41	te-1/1/41	10Gb/s and 1Gb/s
42	te-1/1/42	10Gb/s and 1Gb/s
43	te-1/1/43	10Gb/s and 1Gb/s
44	te-1/1/44	10Gb/s and 1Gb/s
45	te-1/1/45	10Gb/s and 1Gb/s
46	te-1/1/46	10Gb/s and 1Gb/s
47	te-1/1/47	10Gb/s and 1Gb/s
48	te-1/1/48	10Gb/s and 1Gb/s
49	qe-1/1/49	40Gb/s
50	qe-1/1/50	40Gb/s
51	qe-1/1/51	40Gb/s
52	qe-1/1/52	40Gb/s

max(64*10G)

When ports are in max mode, the mapping between physical port , the logical ports/interface names and interfaces support speed is in the following table.

Physical Port number	interface name	interface support speed
1	te-1/1/1	10Gb/s and 1Gb/s
2	te-1/1/2	10Gb/s and 1Gb/s
3	te-1/1/3	10Gb/s and 1Gb/s
4	te-1/1/4	10Gb/s and 1Gb/s
5	te-1/1/5	10Gb/s and 1Gb/s

6	te-1/1/6	10Gb/s and 1Gb/s
7	te-1/1/7	10Gb/s and 1Gb/s
8	te-1/1/8	10Gb/s and 1Gb/s
9	te-1/1/9	10Gb/s and 1Gb/s
10	te-1/1/10	10Gb/s and 1Gb/s
11	te-1/1/11	10Gb/s and 1Gb/s
12	te-1/1/12	10Gb/s and 1Gb/s
13	te-1/1/13	10Gb/s and 1Gb/s
14	te-1/1/14	10Gb/s and 1Gb/s
15	te-1/1/15	10Gb/s and 1Gb/s
16	te-1/1/16	10Gb/s and 1Gb/s
17	te-1/1/17	10Gb/s and 1Gb/s
18	te-1/1/18	10Gb/s and 1Gb/s
19	te-1/1/19	10Gb/s and 1Gb/s
20	te-1/1/20	10Gb/s and 1Gb/s
21	te-1/1/21	10Gb/s and 1Gb/s
22	te-1/1/22	10Gb/s and 1Gb/s
23	te-1/1/23	10Gb/s and 1Gb/s
24	te-1/1/24	10Gb/s and 1Gb/s
25	te-1/1/25	10Gb/s and 1Gb/s
26	te-1/1/26	10Gb/s and 1Gb/s
27	te-1/1/27	10Gb/s and 1Gb/s

28	te-1/1/28	10Gb/s and 1Gb/s
29	te-1/1/29	10Gb/s and 1Gb/s
30	te-1/1/30	10Gb/s and 1Gb/s
31	te-1/1/31	10Gb/s and 1Gb/s
32	te-1/1/32	10Gb/s and 1Gb/s
33	te-1/1/33	10Gb/s and 1Gb/s
34	te-1/1/34	10Gb/s and 1Gb/s
35	te-1/1/35	10Gb/s and 1Gb/s
36	te-1/1/36	10Gb/s and 1Gb/s
37	te-1/1/37	10Gb/s and 1Gb/s
38	te-1/1/38	10Gb/s and 1Gb/s
39	te-1/1/39	10Gb/s and 1Gb/s
40	te-1/1/40	10Gb/s and 1Gb/s
41	te-1/1/41	10Gb/s and 1Gb/s
42	te-1/1/42	10Gb/s and 1Gb/s
43	te-1/1/43	10Gb/s and 1Gb/s
44	te-1/1/44	10Gb/s and 1Gb/s
45	te-1/1/45	10Gb/s and 1Gb/s
46	te-1/1/46	10Gb/s and 1Gb/s
47	te-1/1/47	10Gb/s and 1Gb/s
48	te-1/1/48	10Gb/s and 1Gb/s
49	4 x 10G	

	te-1/1/49	10Gb/s
	te-1/1/50	10Gb/s
	te-1/1/51	10Gb/s
	te-1/1/52	10Gb/s
50	4 x 10G	
	te-1/1/53	10Gb/s
	te-1/1/54	10Gb/s
	te-1/1/55	10Gb/s
	te-1/1/56	10Gb/s
51	4 x 10G	
	te-1/1/57	10Gb/s
	te-1/1/58	10Gb/s
	te-1/1/59	10Gb/s
	te-1/1/60	10Gb/s
52	4 x 10G	
	te-1/1/61	10Gb/s
	te-1/1/62	10Gb/s
	te-1/1/63	10Gb/s
	te-1/1/64	10Gb/s

40G Changes to 4*10G in OVS mode on P-3920

In OVS mode Configuration

You can set the port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  set-qe-port-mode  [normal | max]
```

After setting pors to different mode, it is mandatory to restart the OVS service in order to make the new state to take effect.

```
admin@PicOS-OVS$sudo  service  picos  restart
```

You can take a look of the current port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  show-qe-port-mode
```

normal(48*10G+4*40G)

When ports are in normal mode, the mapping between physical port , interface names and interfaces support speed is in the following table.

Physical Port number	interface name	interface support speed
1	te-1/1/1	10Gb/s and 1Gb/s
2	te-1/1/2	10Gb/s and 1Gb/s
3	te-1/1/3	10Gb/s and 1Gb/s
4	te-1/1/4	10Gb/s and 1Gb/s
5	te-1/1/5	10Gb/s and 1Gb/s
6	te-1/1/6	10Gb/s and 1Gb/s
7	te-1/1/7	10Gb/s and 1Gb/s
8	te-1/1/8	10Gb/s and 1Gb/s
9	te-1/1/9	10Gb/s and 1Gb/s
10	te-1/1/10	10Gb/s and 1Gb/s
11	te-1/1/11	10Gb/s and 1Gb/s
12	te-1/1/12	10Gb/s and 1Gb/s

13	te-1/1/13	10Gb/s and 1Gb/s
14	te-1/1/14	10Gb/s and 1Gb/s
15	te-1/1/15	10Gb/s and 1Gb/s
16	te-1/1/16	10Gb/s and 1Gb/s
17	te-1/1/17	10Gb/s and 1Gb/s
18	te-1/1/18	10Gb/s and 1Gb/s
19	te-1/1/19	10Gb/s and 1Gb/s
20	te-1/1/20	10Gb/s and 1Gb/s
21	te-1/1/21	10Gb/s and 1Gb/s
22	te-1/1/22	10Gb/s and 1Gb/s
23	te-1/1/23	10Gb/s and 1Gb/s
24	te-1/1/24	10Gb/s and 1Gb/s
25	te-1/1/25	10Gb/s and 1Gb/s
26	te-1/1/26	10Gb/s and 1Gb/s
27	te-1/1/27	10Gb/s and 1Gb/s
28	te-1/1/28	10Gb/s and 1Gb/s
29	te-1/1/29	10Gb/s and 1Gb/s
30	te-1/1/30	10Gb/s and 1Gb/s
31	te-1/1/31	10Gb/s and 1Gb/s
32	te-1/1/32	10Gb/s and 1Gb/s
33	te-1/1/33	10Gb/s and 1Gb/s
34	te-1/1/34	10Gb/s and 1Gb/s

35	te-1/1/35	10Gb/s and 1Gb/s
36	te-1/1/36	10Gb/s and 1Gb/s
37	te-1/1/37	10Gb/s and 1Gb/s
38	te-1/1/38	10Gb/s and 1Gb/s
39	te-1/1/39	10Gb/s and 1Gb/s
40	te-1/1/40	10Gb/s and 1Gb/s
41	te-1/1/41	10Gb/s and 1Gb/s
42	te-1/1/42	10Gb/s and 1Gb/s
43	te-1/1/43	10Gb/s and 1Gb/s
44	te-1/1/44	10Gb/s and 1Gb/s
45	te-1/1/45	10Gb/s and 1Gb/s
46	te-1/1/46	10Gb/s and 1Gb/s
47	te-1/1/47	10Gb/s and 1Gb/s
48	te-1/1/48	10Gb/s and 1Gb/s
49	qe-1/1/49	40Gb/s
50	qe-1/1/50	40Gb/s
51	qe-1/1/51	40Gb/s
52	qe-1/1/52	40Gb/s

max(64*10G)

When ports are in max mode, the mapping between physical port , the logical ports/interface names and interfaces support speed is in the following table.

Physical Port number	interface name	interface support speed

1	te-1/1/1	10Gb/s and 1Gb/s
2	te-1/1/2	10Gb/s and 1Gb/s
3	te-1/1/3	10Gb/s and 1Gb/s
4	te-1/1/4	10Gb/s and 1Gb/s
5	te-1/1/5	10Gb/s and 1Gb/s
6	te-1/1/6	10Gb/s and 1Gb/s
7	te-1/1/7	10Gb/s and 1Gb/s
8	te-1/1/8	10Gb/s and 1Gb/s
9	te-1/1/9	10Gb/s and 1Gb/s
10	te-1/1/10	10Gb/s and 1Gb/s
11	te-1/1/11	10Gb/s and 1Gb/s
12	te-1/1/12	10Gb/s and 1Gb/s
13	te-1/1/13	10Gb/s and 1Gb/s
14	te-1/1/14	10Gb/s and 1Gb/s
15	te-1/1/15	10Gb/s and 1Gb/s
16	te-1/1/16	10Gb/s and 1Gb/s
17	te-1/1/17	10Gb/s and 1Gb/s
18	te-1/1/18	10Gb/s and 1Gb/s
19	te-1/1/19	10Gb/s and 1Gb/s
20	te-1/1/20	10Gb/s and 1Gb/s
21	te-1/1/21	10Gb/s and 1Gb/s
22	te-1/1/22	10Gb/s and 1Gb/s

23	te-1/1/23	10Gb/s and 1Gb/s
24	te-1/1/24	10Gb/s and 1Gb/s
25	te-1/1/25	10Gb/s and 1Gb/s
26	te-1/1/26	10Gb/s and 1Gb/s
27	te-1/1/27	10Gb/s and 1Gb/s
28	te-1/1/28	10Gb/s and 1Gb/s
29	te-1/1/29	10Gb/s and 1Gb/s
30	te-1/1/30	10Gb/s and 1Gb/s
31	te-1/1/31	10Gb/s and 1Gb/s
32	te-1/1/32	10Gb/s and 1Gb/s
33	te-1/1/33	10Gb/s and 1Gb/s
34	te-1/1/34	10Gb/s and 1Gb/s
35	te-1/1/35	10Gb/s and 1Gb/s
36	te-1/1/36	10Gb/s and 1Gb/s
37	te-1/1/37	10Gb/s and 1Gb/s
38	te-1/1/38	10Gb/s and 1Gb/s
39	te-1/1/39	10Gb/s and 1Gb/s
40	te-1/1/40	10Gb/s and 1Gb/s
41	te-1/1/41	10Gb/s and 1Gb/s
42	te-1/1/42	10Gb/s and 1Gb/s
43	te-1/1/43	10Gb/s and 1Gb/s
44	te-1/1/44	10Gb/s and 1Gb/s

45	te-1/1/45	10Gb/s and 1Gb/s
46	te-1/1/46	10Gb/s and 1Gb/s
47	te-1/1/47	10Gb/s and 1Gb/s
48	te-1/1/48	10Gb/s and 1Gb/s
49	4 x 10G	
	te-1/1/49	10Gb/s
	te-1/1/50	10Gb/s
	te-1/1/51	10Gb/s
	te-1/1/52	10Gb/s
50	4 x 10G	
	te-1/1/53	10Gb/s
	te-1/1/54	10Gb/s
	te-1/1/55	10Gb/s
	te-1/1/56	10Gb/s
51	4 x 10G	
	te-1/1/57	10Gb/s
	te-1/1/58	10Gb/s
	te-1/1/59	10Gb/s
	te-1/1/60	10Gb/s
52	4 x 10G	
	te-1/1/61	10Gb/s
	te-1/1/62	10Gb/s

	te-1/1/63	10Gb/s
	te-1/1/64	10Gb/s

40G Changes to 4*10G in OVS mode on as5712_54x

In OVS mode Configuration

You can set the port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  set-qe-port-mode  [normal | max]
```

After setting pors to different mode, it is mandatory to restart the OVS service in order to make the new state to take effect.

```
admin@PicOS-OVS$sudo  service  picos  restart
```

You can take a look of the current port mode by issuing:

```
admin@PicOS-OVS$ovs-vsctl  show-qe-port-mode
```

noraml (48*10G+6*40G)

When ports are in normal mode, the mapping between physical port , interface names and interfaces support speed is in the following table.

Physical Port number	interface name	interface support speed
1	te-1/1/1	10Gb/s and 1Gb/s
2	te-1/1/2	10Gb/s and 1Gb/s
3	te-1/1/3	10Gb/s and 1Gb/s
4	te-1/1/4	10Gb/s and 1Gb/s
5	te-1/1/5	10Gb/s and 1Gb/s
6	te-1/1/6	10Gb/s and 1Gb/s
7	te-1/1/7	10Gb/s and 1Gb/s

8	te-1/1/8	10Gb/s and 1Gb/s
9	te-1/1/9	10Gb/s and 1Gb/s
10	te-1/1/10	10Gb/s and 1Gb/s
11	te-1/1/11	10Gb/s and 1Gb/s
12	te-1/1/12	10Gb/s and 1Gb/s
13	te-1/1/13	10Gb/s and 1Gb/s
14	te-1/1/14	10Gb/s and 1Gb/s
15	te-1/1/15	10Gb/s and 1Gb/s
16	te-1/1/16	10Gb/s and 1Gb/s
17	te-1/1/17	10Gb/s and 1Gb/s
18	te-1/1/18	10Gb/s and 1Gb/s
19	te-1/1/19	10Gb/s and 1Gb/s
20	te-1/1/20	10Gb/s and 1Gb/s
21	te-1/1/21	10Gb/s and 1Gb/s
22	te-1/1/22	10Gb/s and 1Gb/s
23	te-1/1/23	10Gb/s and 1Gb/s
24	te-1/1/24	10Gb/s and 1Gb/s
25	te-1/1/25	10Gb/s and 1Gb/s
26	te-1/1/26	10Gb/s and 1Gb/s
27	te-1/1/27	10Gb/s and 1Gb/s
28	te-1/1/28	10Gb/s and 1Gb/s
29	te-1/1/29	10Gb/s and 1Gb/s

30	te-1/1/30	10Gb/s and 1Gb/s
31	te-1/1/31	10Gb/s and 1Gb/s
32	te-1/1/32	10Gb/s and 1Gb/s
33	te-1/1/33	10Gb/s and 1Gb/s
34	te-1/1/34	10Gb/s and 1Gb/s
35	te-1/1/35	10Gb/s and 1Gb/s
36	te-1/1/36	10Gb/s and 1Gb/s
37	te-1/1/37	10Gb/s and 1Gb/s
38	te-1/1/38	10Gb/s and 1Gb/s
39	te-1/1/39	10Gb/s and 1Gb/s
40	te-1/1/40	10Gb/s and 1Gb/s
41	te-1/1/41	10Gb/s and 1Gb/s
42	te-1/1/42	10Gb/s and 1Gb/s
43	te-1/1/43	10Gb/s and 1Gb/s
44	te-1/1/44	10Gb/s and 1Gb/s
45	te-1/1/45	10Gb/s and 1Gb/s
46	te-1/1/46	10Gb/s and 1Gb/s
47	te-1/1/47	10Gb/s and 1Gb/s
48	te-1/1/48	10Gb/s and 1Gb/s
49	qe-1/1/49	40Gb/s
50	qe-1/1/50	40Gb/s
51	qe-1/1/51	40Gb/s

52	qe-1/1/52	40Gb/s
53	qe-1/1/53	40Gb/s
54	qe-1/1/54	40Gb/s

max (72*10G)

When ports are in max mode, the mapping between physical port , the logical ports/interface names and interfaces support speed is in the following table.

Physical Port number	interface name	interface support speed
1	te-1/1/1	10Gb/s and 1Gb/s
2	te-1/1/2	10Gb/s and 1Gb/s
3	te-1/1/3	10Gb/s and 1Gb/s
4	te-1/1/4	10Gb/s and 1Gb/s
5	te-1/1/5	10Gb/s and 1Gb/s
6	te-1/1/6	10Gb/s and 1Gb/s
7	te-1/1/7	10Gb/s and 1Gb/s
8	te-1/1/8	10Gb/s and 1Gb/s
9	te-1/1/9	10Gb/s and 1Gb/s
10	te-1/1/10	10Gb/s and 1Gb/s
11	te-1/1/11	10Gb/s and 1Gb/s
12	te-1/1/12	10Gb/s and 1Gb/s
13	te-1/1/13	10Gb/s and 1Gb/s
14	te-1/1/14	10Gb/s and 1Gb/s
15	te-1/1/15	10Gb/s and 1Gb/s
16	te-1/1/16	10Gb/s and 1Gb/s

17	te-1/1/17	10Gb/s and 1Gb/s
18	te-1/1/18	10Gb/s and 1Gb/s
19	te-1/1/19	10Gb/s and 1Gb/s
20	te-1/1/20	10Gb/s and 1Gb/s
21	te-1/1/21	10Gb/s and 1Gb/s
22	te-1/1/22	10Gb/s and 1Gb/s
23	te-1/1/23	10Gb/s and 1Gb/s
24	te-1/1/24	10Gb/s and 1Gb/s
25	te-1/1/25	10Gb/s and 1Gb/s
26	te-1/1/26	10Gb/s and 1Gb/s
27	te-1/1/27	10Gb/s and 1Gb/s
28	te-1/1/28	10Gb/s and 1Gb/s
29	te-1/1/29	10Gb/s and 1Gb/s
30	te-1/1/30	10Gb/s and 1Gb/s
31	te-1/1/31	10Gb/s and 1Gb/s
32	te-1/1/32	10Gb/s and 1Gb/s
33	te-1/1/33	10Gb/s and 1Gb/s
34	te-1/1/34	10Gb/s and 1Gb/s
35	te-1/1/35	10Gb/s and 1Gb/s
36	te-1/1/36	10Gb/s and 1Gb/s
37	te-1/1/37	10Gb/s and 1Gb/s
38	te-1/1/38	10Gb/s and 1Gb/s

39	te-1/1/39	10Gb/s and 1Gb/s
40	te-1/1/40	10Gb/s and 1Gb/s
41	te-1/1/41	10Gb/s and 1Gb/s
42	te-1/1/42	10Gb/s and 1Gb/s
43	te-1/1/43	10Gb/s and 1Gb/s
44	te-1/1/44	10Gb/s and 1Gb/s
45	te-1/1/45	10Gb/s and 1Gb/s
46	te-1/1/46	10Gb/s and 1Gb/s
47	te-1/1/47	10Gb/s and 1Gb/s
48	te-1/1/48	10Gb/s and 1Gb/s
49	4 x 10G	
	te-1/1/49	10Gb/s
	te-1/1/50	10Gb/s
	te-1/1/51	10Gb/s
	te-1/1/52	10Gb/s
50	4 x 10G	
	te-1/1/53	10Gb/s
	te-1/1/54	10Gb/s
	te-1/1/55	10Gb/s
	te-1/1/56	10Gb/s
51	4 x 10G	
	te-1/1/57	10Gb/s

	te-1/1/58	10Gb/s
	te-1/1/59	10Gb/s
	te-1/1/60	10Gb/s
52	4 x 10G	
	te-1/1/61	10Gb/s
	te-1/1/62	10Gb/s
	te-1/1/63	10Gb/s
	te-1/1/64	10Gb/s
53	4 x 10G	
	te-1/1/65	10Gb/s
	te-1/1/66	10Gb/s
	te-1/1/67	10Gb/s
	te-1/1/68	10Gb/s
54	4 x 10G	
	te-1/1/69	10Gb/s
	te-1/1/70	10Gb/s
	te-1/1/71	10Gb/s
	te-1/1/72	10Gb/s

Configuring sFlow v5

PicOS OVS supports sFlow v5. you can configure the sFlow as following:

```
root@PicOS-OVS$ ovs-vsctl  --id=@s create sFlow agent=eth0
target=\"10.10.50.207:9901\" header=128 sampling=64 polling=10 -- set Bridge
br0 sflow=@s
root@PicOS-OVS$
```

In the above example, the parameters are as follows:

```
COLLECTOR_IP=10.10.50.207
COLLECTOR_PORT=9901
AGENT_IP=eth0
HEADER_BYTES=128
SAMPLING_N=64
POLLING_SECS=10
```

List the configuration of sflow:

```
root@PicOS-OVS$ovs-vsctl list sflow
_uuid               : 88d94294-4bb3-44f3-8a12-6055bf458de6
agent               : "eth0"
external_ids        : {}
header              : 128
polling             : 10
sampling            : 64
targets             : ["10.10.50.207:9901"]
root@PicOS-OVS$
```

To delete an sFlow use the clear command as shown in the following example.

```
root@PicOS-OVS$ ovs-vsctl -- clear Bridge br0 sflow
root@PicOS-OVS$
```

Configuring NetFlow

PicOS OVS supports NetFlow. You can configure the NetFlow as follows:

```
root@PicOS-OVS# ovs-vsctl -- set Bridge br0 netflow=@nf -- --id=@nf create
NetFlow targets=\"10.10.50.207:5566\" active-timeout=30
root@PicOS-OVS#
```

In the above command, the parameters are as follows:

```
COLLECTOR_IP=10.10.50.207
COLLECTOR_PORT=5566
ACTIVE_TIMEOUT=30
```

To delete NetFlow, use the clear command as shown in the following example.

```
root@PicOS-OVS# ovs-vsctl -- clear Bridge br0 netflow
```

Configuring Port Mirroring

The following example shows how to configure the Port Mirroring.

```
root@PicOS-OVS# ovs-vsctl -- set bridge br0 mirrors=@m -- --id=@te-1/1/1 get
Port te-1/1/1 -- --id=@te-1/1/2 get Port te-1/1/2 -- --id=@te-1/1/3 get Port
te-1/1/3 -- --id=@m create Mirror name=mymirror
select-dst-port=@te-1/1/1,@te-1/1/2 select-src-port=@te-1/1/1,@te-1/1/2
output-port=@te-1/1/3
root@PicOS-OVS#
```

The above configuration includes ports te-1/1/1, te-1/1/2 and te-1/1/3. The source port are te-1/1/1 and te-1/1/2 (including the ingress and egress), and the output port (monitor port) is te-1/1/3.
The "select-dst-port" means some packets (in switch chip) will go-out from the specified port (egress).
The "select-src-port" means some packets will enter the specified port (ingress).

Deleting the Mirroring

```
root@PicOS-OVS# ovs-vsctl destroy Mirror mymirror -- clear Bridge br0 mirrors
```

Configuring IPv4 OpenFlow

PicOS OVS supports IPv4 flow in open flow.

Creating an IPv4 flow

```
root@PicOS-OVS# ovs-ofctl add-flow br0
dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,in_port=1,dl_type=0x0800,nw_s
ovs-ofctl dump-flows br0
NXST_FLOW reply (xid=0x4):
cookie=0x0, duration=12.758s, table=0, n_packets=0, n_bytes=0,
tcp,in_port=1,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,nw_src=128.1.1.
actions=output:2,output:3,output:4
cookie=0x0, duration=2180.111s, table=0, n_packets=0, n_bytes=0, priority=0
actions=NORMAL
root@PicOS-OVS#
```

Deleting an IPv4 flow

```
root@PicOS-OVS# ovs-ofctl del-flows br0
dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,in_port=1,dl_type=0x0800,nw_s
```

Removing all flows

```
root@PicOS-OVS# ovs-ofctl del-flows br0
root@XorPlus
```

Configure GRE Tunneling

PicOS OVS supports IP GRE tunnel.

Creating a GRE tunnel

```
root@PicOS-OVS# ovs-vsctl add-port br0 gre1 -- set Interface gre1
type=pica8_gre options:remote_ip=10.10.60.10 options:local_ip=10.10.61.10
options:vlan=1 options:src_mac=00:11:11:11:11:11
options:dst_mac=00:22:22:22:22:22 options:egress_port=ge-1/1/5
```

If you want to create a GRE tunnel, you will need to configure a GRE tunnel along with two flows which are used for sending traffic to the GRE and sending output from the GRE respectively.

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:3073
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=5,actions=mod_dl_src:00:11:11:11:11:11,mod_dl_dst:00:33:33:33:33:33,out
```

The GRE port number starts from 3073, which is the port number of GRE1. The first flow in the above example is configured so that all traffic from port ge-1/1/1 will be sent to GRE tunnel whose port number is 3073. The second flow is configured so that all the traffic coming out from GRE tunnel will be forwarded to port ge-1/1/1 and modify the source MAC address to switch's MAC address and the destination MAC address to the MAC address of the internal target.(If you not specify the dl_src in the action of second flow,then the src mac will be the switch's mac,but the dst mac must be specified.)

Configuring MPLS

The basic MPLS actions are Push, Swap and Pop. Beginning with PicOS 2.4, you do not need to configure the ***dl_src***. The packet **src_mac** pushes MPLS to the MAC address of this switch.
You can add flows to modify and copy the MPLS TTL and IP TTL.
You can push/pop 2 MPLS labels per flow.

Every un-tagged packet is tagged with the default VLAN-ID before Push, Pop and Swap.

Hardware or Software based Forwarding

The flow is pre-installed in hardware if there is enough information on the Flow to be processed by the ASIC.

Here is the minimal set of information needed to process the packet on hardware only:

If the Flow action is a POP : dl_dst, vlan_id, mpls_lse
If the Flow action is a PUSH : dl_dst, vlan_id, dl_type,mpls_lse(only being needed when dl_type is 0x8847 or 0x8848)
If the Flow action is a SWAP :dl_dst, vlan_id, mpls_lse

> There is one exception, if the action is "pop_mpls:0x8847" and match enough fields (dl_dst,vlan_id,mpls_label), the flow becomes a direct flow (hardware only), if the action is "pop_mpls:0x0800", the flow is packet-driven flow.

If there is some information missing to process the packet, the flow is becoming a "packet-driven" flow. It means that the first MPLS packet is sent to CPU which analyse it and download a new flow on hardware with the missing information to handle the packet in hardware. Following packets for this specific flows will then be handled by hardware (ASIC) without reaching the Switch CPU.

1.PUSH MPLS:

Pushing an MPLS Label

In the following configuration, you specify a flow, which should match:
{ in_port=1,dl_type=0x0800, dl_dst=22:00:00:00:00:00,dl_vlan=1}
The action is to push an MPLS label (i.e. 10) and forward to port te-1/1/2
Note that MPLS TTL will copy from the IP header and decrease

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_type=0x0800,dl_dst=22:00:00:00:00:00,dl_vlan=1,actions=push_mpls:0
```

Pushing two MPLS Labels

In the following configuration, specify a flow, which should match { in_port=1,dl_type=0x0800,dl_dst=22:00:00:00:00:00,dl_vlan=1}, the action is to push two labels (i.e. 10 and 20) and forward to port te-1/1/2

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_type=0x0800,dl_dst=22:00:00:00:00:00,dl_vlan=1,actions=push_mpls:0
```

2.SWAP MPLS Label:

Swapping MPLS Labels

In following configuration, you specify a flow, which should match { in_port=1,dl_type=0x8847,dl_dst=22:00:00:00:00:00,dl_vlan=1,mpls_label=10}, the action is to swap label 10 with 20 and forward to port te-1/1/2

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_type=0x8847,dl_dst=22:00:00:00:00:00,dl_vlan=1,mpls_label=10,actio
```

3.POP MPLS Label:

Popping an MPLS Label of the flow

In following configuration, specify a flow, which should match { in_port=1,dl_type=0x8847,dl_dst=22:00:00:00:00:00,dl_vlan=1,mpls_label=10}, the action is to pop MPLS label and forward to port te-1/1/2
Note that MPLS TTL will be copied to IP header TTL and decremented by 1.

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_type=0x8847,dl_dst=22:00:00:00:00:00,dl_vlan=1,mpls_label=10,actio
```

Popping one MPLS Label for flows with Two MPLS Labels

In the following configuration, specify a flow that has two MPLS labels (i.e. 10 and 20). The pop action is always popping the outer MPLS header.
Note that you two label flow is popped only one label, the output packet is also a MPLS packet. Thus, the "pop_mpls:0x8847" must be configured.

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_type=0x8847,dl_dst=22:00:00:00:00:00,dl_vlan=1,mpls_label=10,actio
```

Popping two MPLS Labels for flows with two MPLS Lables

In following configuration, specify a flow which has two labels to pop. The output flow is IP packet. Configure two pop entries to pop the flow.

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_type=0x8847,dl_dst=22:00:00:00:00:00,dl_vlan=1,actions=pop_mpls:0x
```

4.PUSH MPLS Label and VLAN:

Pushing one MPLS Label and one vlan

In following configuration, specify flows to push one MPLS Label and one VLAN

```
ovs-ofctl add-flow br0
in_port=2,dl_type=0x0800,dl_vlan=2999,dl_dst=22:22:22:22:22:22,actions=push_mpl
add-flow br0
in_port=2,dl_type=0x8847,dl_vlan=2999,dl_dst=22:22:22:22:22:22,mpls_label=66,ac
```

Pushing two MPLS Labels and one VLAN

In following configuration, specify flows to push two mpls labels and one VLAN

```
ovs-ofctl add-flow br0
in_port=2,dl_type=0x0800,dl_vlan=2999,actions=push_mpls:0x8847,set_field:333-\>
```

5.POP One or Two MPLS Labels:

In following configuration, specify flows which should match dl_type,dl_vlan and mpls label and action is to pop one mpls label and set new src mac address. The first flow will pop one mpls label and the second flow will pop two mpls labels.

```
ovs-ofctl add-flow br0
in_port=2,dl_type=0x8847,dl_vlan=2999,mpls_label=333,dl_dst=22:22:22:22:22:22,a
add-flow br0
in_port=2,dl_type=0x8847,dl_vlan=2999,actions=pop_mpls:0x0800,output:4
```

6.POP One or Two MPLS Labels and PUSH VLAN:

The following flow should match dl_type and dl_vlan, action is to pop one mpls label and push one vlan.

```
ovs-ofctl add-flow br0
in_port=2,dl_type=0x8847,dl_vlan=2999,actions=pop_mpls:0x8847,push_vlan:0x8100,
```

The following flow should match dl_type and dl_vlan, with action of popping two mpls labels and pushing one vlan.

```
ovs-ofctl add-flow br0
in_port=2,dl_type=0x8847,dl_vlan=2999,actions=pop_mpls:0x8847,pop_mpls:0x8847,p
```

The following flow should match dl_type and dl_vlan and mpls label, with action is to pop two mpls labels and push one vlan.

```
ovs-ofctl add-flow br0
in_port=2,dl_type=0x8847,dl_vlan=2999,mpls_label=333,actions=pop_mpls:0x0800,pu
```

The following flow should match dl_type, with action is to pop two mpls labels and push one vlan.

```
ovs-ofctl add-flow br0
in_port=2,dl_type=0x8847,actions=pop_mpls:0x8847,push_vlan:0x8100,set_field:199
```

7.PUSH MPLS and POP VLAN:

In following configuration, push mpls label and pop vlan have been supported. As following flow, action is to push one mpls label and pop one vlan.

```
ovs-ofctl add-flow br0
in_port=2,dl_type=0x8847,dl_vlan=2999,actions=push_mpls:0x8847,set_field:333-\>
```

8.PUSH Two MPLS Labels and POP VLAN:

As following flow, action is to push two mpls labels and pop vlan.

```
ovs-ofctl add-flow br0
in_port=2,dl_type=0x8847,dl_vlan=2999,actions=push_mpls:0x8847,set_field:333-\>
```

9.PUSH Two MPLS Labels and POP Two VLANs:

As following flow, actions is to push two mpls labels and pop two vlans.

```
ovs-ofctl add-flow br0
in_port=1,dl_type=0x8847,dl_vlan=1666,actions=push_mpls:0x8847,set_field:333-\>
```

NOTICE:

1.If flows match dl_vlan and the actions have push_vlan, then receives packets only carry the pushed vlan.

2.Hardware can’t support pop_mpls and pop_vlan at the same time, and the packets can’t forward with line-speed.

3.When actions don’t appoint to modify dl_src, the src mac of received packets should be modified to bridge mac whatever for direct flows or packet-driven flow.

4.Push two mpls labels is supported, but push two vlans at the same time is not supported.

Configuring LAG and LACP

PicOS OVS supports LAG and LACP
PicOS can support 48 LAG or LACP at most. Each LAG has 8 member ports at most

Create a static LAG

In following configuration, you can create LAG ae1, and add port 2 and port 3 into this LAG

```
root@PicOS-OVS# ovs-vsctl  add-port br0 ae1 vlan_mode=trunk tag=1 -- set
Interface ae1 type=pica8_lag
root@PicOS-OVS# ovs-vsctl -- set Interface ae1 options:lag_type=static
root@PicOS-OVS# ovs-vsctl -- set Interface ae1
options:members=ge-1/1/2,ge-1/1/3
```

Create a LACP port

In following configuration, you create a LACP port and configure the parameter

```
root@PicOS-OVS# ovs-vsctl  add-port br0 ae1 vlan_mode=trunk tag=1 -- set
Interface ae1 type=pica8_lag
root@PicOS-OVS# ovs-vsctl  -- set Interface ae1 options:lag_type=lacp
root@PicOS-OVS# ovs-vsctl  -- set Interface ae1
options:members=ge-1/1/2,ge-1/1/3
root@PicOS-OVS# ovs-vsctl  -- set Interface ae1
options:lacp-system-id=00:11:11:11:11:11
root@PicOS-OVS# ovs-vsctl  -- set Interface ae1
options:lacp-system-priority=32768
root@PicOS-OVS# ovs-vsctl  -- set Interface ae1 options:lacp-time=fast
root@PicOS-OVS# ovs-vsctl  -- set Interface ae1 options:lacp-time=slow
root@PicOS-OVS# ovs-vsctl  -- set Interface ae1 options:lacp-mode=active
root@PicOS-OVS# ovs-vsctl  -- set Interface ae1 options:lacp-mode=passive
root@PicOS-OVS# ovs-vsctl  -- set Interface ge-1/1/2 options:lacp-port-id=2
root@PicOS-OVS# ovs-vsctl  -- set Interface ge-1/1/2
options:lacp-port-priority=32768
root@PicOS-OVS# ovs-vsctl  -- set Interface ge-1/1/2
options:lacp-aggregation-key=0
```

Create static flow for LAG or LACP

In following configuration, you can create static flow whose output port is LAG or LACP.

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1025,actions=output:1
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:1025
```

LAG number index is shown as following:

For all the switch, lag number index is as follow.

---

lag name	ae1	ae2	...	ae1023

---

lag number index	1025	1026	...	2047

Display the information of LACP

You can display the information of LACP with following CLI.

```
root@PicOS-OVS# ovs-appctl -t ovs-vswitchd lacp/show
```

Lag hash config

Lag hash commands are as follow.

```
# Config command
ovs-vsctl -- set Interface ae1 options:hash-mapping=dl_dst
ovs-vsctl -- set Interface ae1 options:hash-mapping=dl_src_dst
ovs-vsctl -- set Interface ae1 options:hash-mapping=dl_src
ovs-vsctl -- set Interface ae1 options:hash-mapping=nw_dst
ovs-vsctl -- set Interface ae1 options:hash-mapping=nw_src_dst
ovs-vsctl -- set Interface ae1 options:hash-mapping=nw_src
ovs-vsctl -- set Interface ae1 options:hash-mapping=resilient
ovs-vsctl -- set Interface ae1 options:hash-mapping=advance
ovs-vsctl -- set-lag-advance-hash-mapping-fields dl_dst dl_src ether_type
in_port nw_dst nw_proto nw_src port_dst port_src
vlan
# Show command
ovs-vsctl show-lag-advance-hash-mapping-fields
```

Creating a Group Table

PicOS OVS supports group table in Openflow 1.2

Because of the ASIC limitation, not all buckets in a group table will be installed to ASIC for a flow. The system will install buckets at most as possible to ASIC.

Create group table

In following configuration, create a group table and a flow whose action is a group table

type=all

```
root@PicOS-OVS# ovs-ofctl add-group br0
group_id=2238,type=all,bucket=output:2
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,dl_type=0x0800,nw_p
```

type=indirect

```
root@PicOS-OVS# ovs-ofctl add-group br0
group_id=2239,type=indirect,bucket=output:2
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,dl_type=0x0800,nw_p
```

type=fast_failover

```
root@PicOS-OVS# ovs-ofctl add-group br0 group_id=2,type=all,bucket=output:2
root@PicOS-OVS# ovs-ofctl add-group br0 group_id=3,type=all,bucket=output:3
root@PicOS-OVS# ovs-ofctl add-group br0
group_id=4,type=fast_failover,bucket=watch_port:2,watch_group:2,output:4,watch_
ovs-ofctl add-flow br0
in_port=1,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,dl_type=0x0800,nw_p
```

Modify bucket in a group table

In following configuration, you are modifying the buckets in a group table

```
root@PicOS-OVS# ovs-ofctl mod-group br0
group_id=2238,type=all,bucket=output:3
root@PicOS-OVS# ovs-ofctl mod-group br0
group_id=2238,type=all,bucket=output:2,bucket=output:3
root@PicOS-OVS# ovs-ofctl mod-group br0
group_id=2238,type=all,bucket=mod_dl_src:22:11:11:22:22:22,mod_dl_dst:22:00:00:
```

Delete group table

In following configuration, you can delete the group table with following CLI.

```
root@PicOS-OVS# ovs-ofctl del-groups br0 group_id=2238
```

Display the information of group table

Use can display the information of all group table.

```
root@PicOS-OVS# ovs-ofctl dump-groups br0
root@PicOS-OVS# ovs-ofctl dump-group-stats br0 group_id=2238
root@PicOS-OVS# ovs-ofctl dump-group-stats br0 group_id=all
root@PicOS-OVS# ovs-ofctl dump-group-features br0
```

Configuring ECMP

Command

ovs-vsctl set-max-ecmp-ports *[numbers]*

ovs-vsctl show-max-ecmp-ports

Parameters

numbers:[2-32],max is 32 ,and this must be 2^n(n=1,2,3,4,5)

default *number is 4.*

Example

PicOS OVS supports ecmp (nw_src, nw_dst),the default ecmp ports is 4.
Ip packets (nw_src=192.168.1.0/255.255.255.1) will forward to port 2.
Ip packets (nw_src=192.168.1.1/255.255.255.1) will forward to port 3.

```
root@PicOS-OVS#ovs-ofctl add-group br0
group_id=1,type=select,bucket=output:2,bucket=output:3
root@PicOS-OVS#ovs-ofctl add-flow br0
dl_type=0x0800,nw_src=192.168.1.0/24,actions=group:1
```

If port 2 is down, all packets will forward to port 3.

Class of Service Mapping for QoS

In PicOS-2.1, if you enable the Class of Service (CoS) mapping, the packet mapped to a physical queues (0-7). With DSCP (0-7), it maps to queue-0 and with DSCP (8-16), it maps to queue-1 and so on. Queue-7 has the highest priority. Enable the CoS Mapping as following:

```
root@PicOS-OVS#ovs-vsctl set-cos-map true
```

Display the configuration by following:

```
root@PicOS-OVS#ovs-vsctl show-cos-map
```

If you want to configures a flow, use the following command:

```
root@PicOS-OVS#ovs-ofctl add-flow br0
in_port=1,dl_src=22:11:11:11:11:11,actions=set_queue:7,output=3
```

The action of "set-queue:7 "will take the place of the default CoS mapping

Configuring QoS Queue

PicOS OVS supports qos/queue
Flow (dl_src is 22:11:11:11:11:11) will be forward to queue 0 of port 3
Flow (dl_src is 22:11:11:11:11:12) will be forward to queue 7 of port 3.
Min and max rate of queue 0 and queue 7 is set as 10M

```
root@PicOS-OVS# ovs-ofctl del-flows br0
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_src=22:11:11:11:11:11,actions=set_queue:0,output=3
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=2,dl_src=22:11:11:11:11:12,actions=set_queue:7,output=3
root@PicOS-OVS# ovs-vsctl -- set port ge-1/1/3 qos=@newqos -- --id=@newqos
create qos type=PRONTO_STRICT queues:0=@newqueue queues:7=@newqueue1 --
--id=@newqueue create queue other-config:min-rate=10000000
other-config:max-rate=10000000 -- --id=@newqueue1 create queue
other-config:min-rate=10000000 other-config:max-rate=10000000
```

Result:
Port 3 receive all packets from port 2, and a little from port 1.
Receive rate of port 3 is about 10Mbps+10Mbps.

In PicOS switch, there are 7 queues in ASIC with priority 0~7 (In current version, P3922/P3780/3920, only support queues 0~4). The queue 7 has the highest priority and queue 0 has the lowest priority. When user configure a flow which will be forwarded in queue 3 (set_queue:3), all packet match this flow will be forwarded in physical queue 3.
“PRONTO_STRICT” means the scheduler is based on Strict priority between queues. In other word, if the high priority queue has packets, the scheduler will never send packet in the low priority queues.

Configuring OpenFlow Meter

PicOS OVS supports meter in Openflow 1.3

Create meter

In the following configuration, you can create a meter. The valid meter ID is from 1 to 100. Define the meter before using it.

type=drop,without burst_size

30M bits per second will be forward to port 2.

```
root@PicOS-OVS# ovs-ofctl add-meter br0
meter=2,kbps,band=type=drop,rate=30000
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=meter:2,output:2
```

type=drop,with burst_size

30M bits per second will be forward to port 2.

```
root@PicOS-OVS# ovs-ofctl add-meter br0
meter=2,kbps,burst,band=type=drop,rate=30000,burst_size=30000
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=meter:2,output:2
```

type=dscp_remark,without burst_size

30M bits per second dscp value is changed as 14.

```
root@PicOS-OVS# ovs-ofctl add-meter br0
meter=2,kbps,band=type=dscp_remark,rate=30000,prec_level=14
```

type=dscp_remark,with burst_size

30M bits per second dscp value is changed as 14.

```
root@PicOS-OVS# ovs-ofctl add-meter br0
meter=2,kbps,burst,band=type=dscp_remark,rate=30000,prec_level=14,burst_size=30
```

Modify meter

In following configuration, you can modify the meter

```
root@PicOS-OVS# ovs-ofctl mod-meter br0
meter=2,kbps,band=type=dscp_remark,rate=30000,prec_level=12
root@PicOS-OVS# ovs-ofctl mod-meter br0
meter=2,kbps,burst,band=type=drop,rate=10000,burst_size=30000
```

Delete meter

In following configuration, you delete the meter

```
root@PicOS-OVS# ovs-ofctl del-meters br0
root@PicOS-OVS# ovs-ofctl del-meter br0 meter=1
```

Display the information of meter

Use can display the information of all meter

```
root@PicOS-OVS# ovs-ofctl meter-features br0
root@PicOS-OVS# ovs-ofctl dump-meters br0
root@PicOS-OVS# ovs-ofctl meter-stats br0
```

Configuring QinQ

PicOS OVS supports qInq. (3290,3295 do not support set inner pcp)

Push tag, Push <tag:2000>

root@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,actions=push_vlan:0x8100,set_field:2000-\>vlan_vid,output:2

Push <tag:2000 pcp:3>

root@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,actions=push_vlan:0x8100,set_field:2000-\>vlan_vid,set_field:3-\>vlan_pcp,output:2

Push <tag:3000 tag:4094>

```
root@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,actions=push_vlan:0x8100,set_field:3000-\>vlan_vid,push_vlan:0x8100,s
```

Push <tag:3000 tag:4094 pcp:3>

```
root@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,actions=push_vlan:0x8100,set_field:3000-\>vlan_vid,push_vlan:0x8100,s
```

Push <tag:3000 pcp:3 tag:4094 pcp:7>

```
root@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,actions=push_vlan:0x8100,set_field:3000-\>vlan_vid,set_field:3-\>vlan
```

Pop tag, Pop one header

```
root@PicOS-OVS$ovs-ofctl add-flow br0 in_port=1,actions=pop_vlan,output:2
```

Pop two header

```
root@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,actions=pop_vlan,pop_vlan,output:2
```

You can also use the strip_vlan to achieve pop VLAN tagged, for example:

```
root@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,priority=100,actions=strip_vlan,output:2
```

In hardware ASIC, the implementation of "strip_vlan" is: change the packet's tag to "4095" and strip the vlan tag of 4095 in the egress. Thus, above flow will be split two flows in ingress and egress respectively as following:
Ingress "in_pot=1, priority=100, action=set_field:2000-\>vlan_vid"
Egress "in_pot=1, priority=100,action=strip_vlan,output:2"
In this case, maybe other traffic which match the egress flow will be stripped vlan and forwarded to port-3. You can install other flow with higher priority to avoid this problem.

Hardware limitation of Pushing Two Tags

There is a limitation in pushing two tags; hardware ASIC can only identify two tags.
a. If primary packet is untagged, do two push_vlan (add tagA, tagB), output packets is tagged with two vlans. (tagA, tagB)
b. If primary packet is tagged with one vlan (tag0), do two push_vlan (add tagA, tagB), output packets is tagged with two vlans. (tag0, tagB),
c. If primary packet is tagged with more than one vlan (outer vlan is tag0), do two push_vlan (add tagA, tagB), output packets is tagged with vlans. (tag0 and other tag of primary packets, tagB),

Configuring OpenFlow Provider Backbone Bridge

PicOS OVS supports pbb in Openflow 1.3, P3297, P3780, P3920, P3922, P5101, P5401 and above switches support this feature.

push

Push pbb_isid,eth_src,eth_dst

Outer src mac is set as 00:00:00:11:11:11, and dsc mac is set as 00:00:00:22:22:22, Vlan is set as 4094, pbb isid is set as 23.

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=11,dl_type=0x0800,dl_src=22:11:11:11:11:11,dl_dst=22:22:22:22:22:22,act
```

Push pbb without pbb_isid,eth_src,eth_dst

Outer src mac is set as 22:11:11:11:11:11, and dsc mac is set as 22:22:22:22:22:22, Vlan is set as 4094, pbb isid is set as 0.

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=11,dl_type=0x0800,dl_src=22:11:11:11:11:11,dl_dst=22:22:22:22:22:22,act
```

Push pbb_isid,eth_src,eth_dst for pbb packets

Outer src mac is set as 00:00:00:11:11:11, and dsc mac is set as 00:00:00:22:22:22,
Vlan is set as 4094, pbb isid is set as 21. (isid of primary pbb packet should not be 21)

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=11,dl_type=0x88e7,actions=push_pbb:0x88e7,set_field:21-\>pbb_isid,set_f
```

pop

Pop pbb packets tagged with vlan 1 (Primary pbb packets should be tagged with vlan 1) Pbb packets are popped.

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=11,dl_type=0x8100,dl_src=00:00:00:11:11:11,dl_dst=00:00:00:22:22:22,act
```

Pop pbb packets tagged with vlan 2000 (Primary pbb packets should be tagged with vlan 2000) Pbb packets are popped.

```
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=11,dl_type=0x8100,dl_vlan=2000,dl_src=00:00:00:11:11:11,dl_dst=00:00:00
```

Important Things to Know

- Push pbb should be done with push_vlan,

- When do push pbb, primary src mac, and dst mac will be used if no config of eth_src , eth_dst
- Do push pbb for pbb packet, primary pbb isid should be not same as the push pb isid.
- When do pop pbb, primary packets should include vlan, and actions should include pop_vlan.

Configuring OpenFlow Loopback

Configure the possibility to have egress interface to be the ingress interface

By default, a packet coming on an interface cannot be sent back to the same interface via Openflow. This behavior can be changed with the following commands:

```
ovs-appctl loopback/enable true
```

This is supported starting in PicOS 2.2. It should only be used for specific traffic as it can be dangerous to send broadcast traffic back on the same port on a Layer 2 network.

Enabling Loopback Interface

PicOS supports Loopback interface in hardware. By default, you cannot configure a flow whose output port is the "in_port". For example, the following flow will not work in hardware by default:

```
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:1
```

Enable these kind of loopback interface by following CLI:

```
root@PicOS-OVS#ovs-appctl loopback/enable true
```

With the above configuration, the flow output port is the same as in_port will work in hardware. You can disable the loopback interface with the following command:

```
ovs-appctl loopback/enable false
```

You should know the limitation of the loopback interface in hardware. In the Openflow Specification, there are some actions (Flood, Group table, for example) that are for broadcasting. The packet should not be forwarded back to the **in_port port**. Be cautious using the enable loopback interface so that the packet is not forwarded back to the in_port port.

Optimizing TCAM Usage

By default, 2 TCAM entries are used to support all matching tuples for all flows even the flow does not use all matching tuples.

PicOS allows you to configure the switch in short flow TCAM match mode to optimize the TCAM usage, in this mode, each flow will only consume 1 TCAM entry (doubling the flows capacity in the TCAM).

When this mode is enabled (with the set-match-mode command), only specific fields can be used in the priority range defined by the command.

The flows must use the exact fields described below:

mac mode: "in_port, dl_src, dl_dst, vlan_vid, dl_type"
ip mode: "in_port, nw_proto, nw_src, nw_dst, dl_type=0x0800"
arp_tpa mode: "in_port, arp_tpa, dl_type=0x0806"

ipv6_full mode: "in_port,dl_vlan,ipv6_src,ipv6_dst,nw_proto,dl_type=0x86dd"

ipv6_src mode: "in_port,dl_src,dl_dst,dl_vlan,ipv6_src,nw_proto,dl_type=0x86dd"

ipv6_dst mode: "in_port,dl_src,dl_dst,dl_vlan,ipv6_dst,nw_proto,dl_type=0x86dd"

For example, if mac mode is enabled, all the flows must only use one or more fields defined in the mac mode. If mac and ip modes are enabled, then you can configure either mac flows or ip flows based on the fields described above. However, you cannot mixed the fields from mac and ip (that is, dl_src and nw_src).

Each mode is configured with a priority range. This range can only be used by the specific type of flows configured.

in the example below, all the Flows between priority "10 and 1000" have to be Mac flows. All flows between 2000 and 20000 have to be IP flows and all flows between 30000 and 60000 have to be ARP flows, all flows between 50000 and 50001 have to be ipv6_full, all flows between 50002 and 50003 have to be ipv6_dst, all flows between 50004 and 60000 have to be ipv6_src.

```
ovs-vsctl set-match-mode
mac=10-1000,ip=2000-20000,arp_tpa=30000-40000,ipv6_full=50000-50001,ipv6_dst=50
```

You can display this configuration with the following command:

```
ovs-vsctl show-match-mode
```

You can remove this configuration with the following command:

```
ovs-vsctl set-match-mode default
```

Once the mode is reconfigured to the default mode or another mode, the current flow table is flushed and start clean.

Flow configuration examples are as follow.

```
ovs-ofctl add-flow br0
priority=900,dl_src=22:11:11:11:11:11,dl_dst=22:00:00:00:00:00,actions=output:2
add-flow br0
priority=2000,nw_src=192.168.1.1,nw_dst=192.168.100.100,nw_proto=6,dl_type=0x08
add-flow br0
priority=30000,arp_tpa=192.168.2.2,dl_type=0x0806,actions=output:2
ovs-ofctl add-flow br0
priority=50000,dl_vlan=1,ipv6_src=2001:2:0:0:0:0:0:0,ipv6_dst=2001:1:0:0:0:0:
add-flow br0
priority=50002,dl_dst=22:00:00:00:00:00,dl_src=22:11:11:11:11:11,dl_vlan=1,ipv6
add-flow br0
priority=50004,dl_dst=22:00:00:00:00:00,dl_src=22:11:11:11:11:11,dl_vlan=1,ipv6
```

From picos2.4, ipv6_full, ipv6_src, ipv6_dst match mode are supported.

Configuring Layer 2 over GRE on 5101 and 5401 Switches

Only switches 5101 and 5401 support Layer 2 over Generic Routing Encapsulation (L2GRE); the port number of L2GRE ranges from 5121 to 6143. GRE is an encapsulated mechanism that encapsulate packet IPs; L2GRE is an encapsulated mechanism that encapsulate the entire packet.To resolve the problem that pushing the interface PVID to the untag packets before encapsulated the L2GRE header, use the command **ovs-vsctl set interface <interface> type=pica8 options:access-vport=true**. Like this, the untag packets can be encapsulated by L2GRE header with no VLAN; that is, the PVID of ingress port. And the tagged packets are encapsulated by L2GRE header, the inner VLAN is the VLAN tag of the packets that are received by ingress port. See the example below.

```
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre1 -- set interface l2gre1
type=pica8_l2gre options:remote_ip=10.10.61.10 options:local_ip=10.10.60.10
options:vlan=1 options:l2gre_key=1234 options:src_mac=C8:0A:A9:9E:49:1A
options:dst_mac=C8:0A:A9:9E:14:A5 options:egress_port=qe-1/1/2
```

Examples

push one L2GRE header

topology

Creating a L2GRE tunnel

(1) create a new bridge named br0.

```
admin@PicOS-OVS$ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
```

(2) add ports to br0.

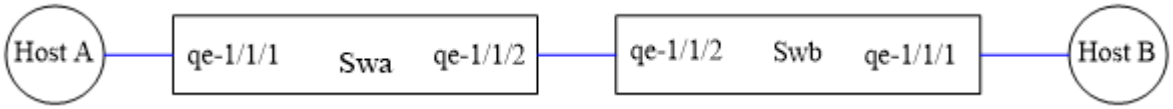
```
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/1 vlan_mode=trunk tag=1 -- set
Interface qe-1/1/1 type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/2 vlan_mode=trunk tag=1 -- set
Interface qe-1/1/2 type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre1 -- set interface l2gre1
type=pica8_l2gre options:remote_ip=10.10.61.10 options:local_ip=10.10.60.10
options:vlan=1 options:l2gre_key=1234 options:src_mac=C8:0A:A9:9E:49:1A
options:dst_mac=C8:0A:A9:9E:14:A5 options:egress_port=qe-1/1/2
```

You must configure a flow if you want to send packets to a L2GRE port. And port number is 5121 for l2gre1 tunnel, different L2GRE tunnel must have different l2gre_key.

```
admin@PicOS-OVS$ovs-ofctl add-flow br0 in_port=1,actions=output:5121
```

Send packets (ARP, L2/L3 packets) to qe-1/1/1,then packets are encapsulated by L2GRE header. when the VLAN tag of Layer 2 GRE tunnel is the same with native VLAN-ID of output port, L2GRE VLAN of the packets are stripped when forwarded by egress port. When the VLAN tag of L2GRE tunnel is different from native VLAN-ID \output port, L2GRE VLAN of the packets are not stripped when forwarded by egress port.

strip L2GRE tunnel



configuration

configure the L2GRE tunnels named l2gre1 on qe-1/1/2 of swa and l2gre2 on qe-1/1/2 of swb.

swa:

```
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre1 -- set Interface l2gre1
type=pica8_l2gre options:remote_ip=10.10.61.10 options:local_ip=10.10.60.10
options:vlan=2 options:l2gre_key=1234 options:src_mac=C8:0A:A9:04:49:1A
options:dst_mac=C8:0A:A9:9E:14:A5  options:egress_port=qe-1/1/2
```

swb:

```
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre1 -- set Interface l2gre1
type=pica8_l2gre options:remote_ip=10.10.60.10 options:local_ip=10.10.61.10
options:vlan=2 options:l2gre_key=1234 options:src_mac=C8:0A:A9:9E:14:A5
options:dst_mac=C8:0A:A9:04:49:1A options:egress_port=qe-1/1/2
```

You must add the two flows below if you want to push L2GRE header on qe-1/1/2 of swa and strip the Layer 2 GRE header on qe-1/1/2 of swb.

Swa:

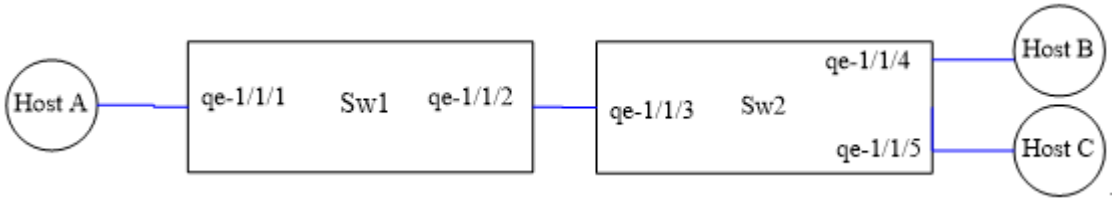
```
admin@PicOS-OVS$ovs-ofctl add-flow br0 in_port=1,actions=output:5121
```

swb:

```
admin@PicOS-OVS$ovs-ofctl add-flow br0 in_port=5121,actions=output:1
```

qe-1/1/1 of swb will receive the original packets (the contents of packets are the same with packets that qe-1/1/1 of swa received).

configure two L2GRE tunnels on one physical port



Configuration

Configure two L2GRE tunnels on both qe-1/1/2(l2gre1,l2gre2) and qe-1/1/3(l2gre1,l2gre2). These two tunnels have different IP and l2gre_key so you must configure some flows.

Sw1:

```
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre1 -- set Interface l2gre1
type=pica8_l2gre options:remote_ip=10.10.60.10 options:local_ip=10.10.61.10
options:vlan=2 options:l2gre_key=1234 options:src_mac=C8:0A:A9:9E:14:A5
options:dst_mac=C8:0A:A9:04:49:1A options:egress_port=qe-1/1/2
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre2 -- set Interface l2gre2
type=pica8_l2gre options:remote_ip="10.10.61.61 options:local_ip=10.10.60.60
options:vlan=10 options:l2gre_key=1235 options:src_mac=C8:0A:A9:04:49:1A
options:dst_mac=88:88:88:88:88:88 options:egress_port=qe-1/1/2"
```

flows in sw1,

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,dl_dst=22:66:66:66:66:66,actions=output:5121
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,dl_dst=22:66:66:66:66:67,actions=output:5122
```

sw2:

```
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre1 -- set Interface l2gre1
type=pica8_l2gre options:remote_ip=10.10.61.10 options:local_ip=10.10.60.10
options:vlan=2 options:l2gre_key=1234 options:src_mac=C8:0A:A9:04:49:1A
options:dst_mac=C8:0A:A9:9E:14:A5 options:egress_port=qe-1/1/3
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre2 -- set Interface l2gre2
type=pica8_l2gre options:remote_ip=10.10.60.60 options:local_ip=10.10.61.61
options:vlan=10 options:l2gre_key=1235 options:src_mac=88:88:88:88:88:88
options:dst_mac=C8:0A:A9:04:49:1A options:egress_port=qe-1/1/3
```

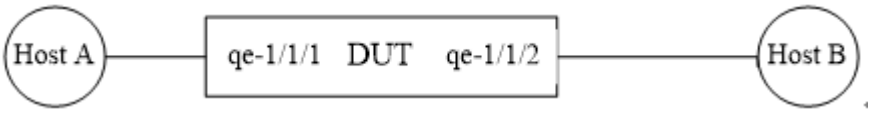
flows in sw2

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=5121,dl_dst=22:66:66:66:66:66,actions=output:4
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=5122,dl_dst=22:66:66:66:66:67,actions=output:5
```

send packets to qe-1/1/1,different packets will go to different L2GRE tunnels. When they are stripped L2GRE header on qe-1/1/3,they are forwarded to a different port.

Length of l2gre_key

In pica8 switch, the length of l2gre_key can be 16bit, 20bit, 24bit or 32bit; 20 bit is the default value.



configuration

configure the L2GRE tunnel on qe-1/1/2 .

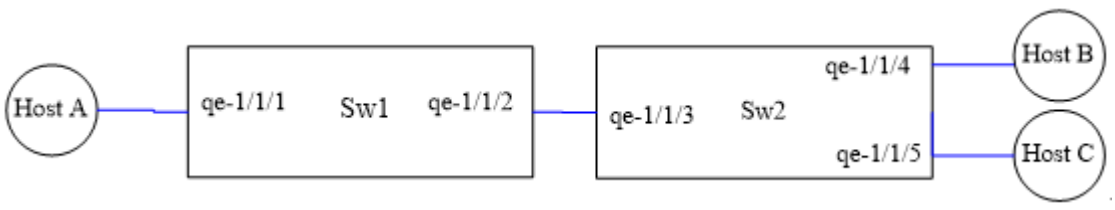
```
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre1 -- set Interface l2gre1
type=pica8_l2gre options:remote_ip=10.10.61.10 options:local_ip=10.10.60.10
options:vlan=1 options:l2gre_key=1234 options:src_mac=C8:0A:A9:04:49:1A
options:dst_mac=C8:0A:A9:9E:14:A5 options:egress_port=qe-1/1/2
```

Add a flow to switch

```
admin@PicOS-OVS$ovs-ofctl add-flow br0 in_port=1,actions=output:5121
```

This key of L2GRE tunnel is 1234 here in decimal, 4d2 in hex. The default value of Layer 2 GRE key is 20, so the value of GRE key of packets is 0x004d2000. When you set the l2gre_key value to 16 using the command **ovs-vsctl set-l2gre-key-length 16,** the value of the GRE key of packet is 0x04d20000. When you set the l2gre_key value to 20 using the command **ovs-vsctl set-l2gre-key-length 24,** the value of GRE key packet is 0x0004d200. When you set the l2gre_key value to 32 using the command **ovs-vsctl set-l2gre-key-length 32,** the value of GRE key packet is 0x000004d2.

Collaboration between nvgre and VXLAN



Configuration

configure the L2GRE tunnel and VXLAN tunnel on qe-1/1/2 and qe-1/1/3.

```
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre1 -- set Interface l2gre1
type=pica8_l2gre options:remote_ip=10.10.61.10 options:local_ip=10.10.60.10
options:vlan=1 options:l2gre_key=1234 options:src_mac=C8:0A:A9:04:49:1A
options:dst_mac=C8:0A:A9:9E:14:A5  options:egress_port=qe-1/1/2
admin@PicOS-OVS$
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre1 -- set Interface l2gre1
type=pica8_l2gre options:remote_ip=10.10.60.10 options:local_ip=10.10.61.10
options:vlan=1 options:l2gre_key=1234 options:src_mac=C8:0A:A9:9E:14:A5
options:dst_mac=C8:0A:A9:04:49:1A  options:egress_port=qe-1/1/3
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1
type=pica8_vxlan options:remote_ip=10.10.10.2 options:local_ip=10.10.10.1
options:vlan=1 options:vnid=1122867 options:udp_dst_port=4789
options:src_mac=66:66:66:77:77:77 options:dst_mac=88:88:88:77:77:77
options:egress_port=qe-1/1/2
admin@PicOS-OVS$
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1
type=pica8_vxlan options:remote_ip=10.10.10.1 options:local_ip=10.10.10.2
options:vlan=1 options:vnid=1122867 options:udp_dst_port=4789
options:src_mac=88:88:88:77:77:77 options:dst_mac=66:66:66:77:77:77
options:egress_port=qe-1/1/3
```

Flows in Switches

sw1:

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,dl_dst=22:22:22:22:22:23,actions=output:4097
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,dl_dst=22:22:22:22:22:22,actions=output:5121
```

sw2:

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=5122,dl_dst=22:22:22:22:22:22,actions=output:4
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=4098,dl_dst=22:22:22:22:22:23,actions=output:5
```

qe-1/1/4 should receive the de-capsulated packets with src_mac 22:22:22:22:22:22, qe-1/1/5 should receive the de-capsulated packets with src_mac 22:22:22:22:22:23.That is to say VXLAN and L2GRE do not affect each other.

Configuring VXLAN

Only switch with ASIC Trident-II (e.g. P5401 and P5101) can support vxlan, and the port number of vxlan ranges from 4097 to 5119. VXLAN mechanism is based on the limited number of vlans(0-4094).To provide more networks for switches or host,vxlan occurs.To resolve the problem that pushing the interface' PVID to the untag packets before encapsulated the VXLAN header,you must use this command "ovs-vsctl set interface <interface> type=pica8 options:access-vport=true ". Like this,the untag packets can be encapsulated by vxlan header with no vlan that is pvid of ingress port.And the tagged packets are encapsulated by vxlan header ,the inner vlan is the vlan tag of packets that received by ingress port.

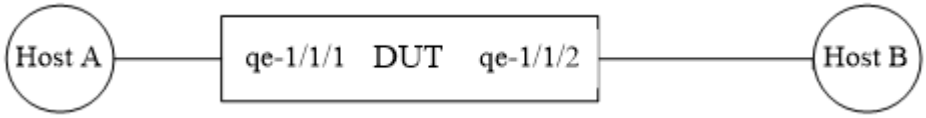
Command

```
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1
type=pica8_vxlan options:remote_ip=10.10.10.2 options:local_ip=10.10.10.1
options:vlan=1 options:vnid=1122867 options:udp_dst_port=4789
options:src_mac=C8:0A:A9:04:49:1A  options:dst_mac=C8:0A:A9:9E:14:A5
options:egress_port=qe-1/1/2
```

Examples

configure a VXLAN tunnel

topology



configuration

(1)create a new bridge named br0.

```
admin@PicOS-OVS$ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
```

(2)add ports to br0.

```
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/1 vlan_mode=trunk tag=1  -- set
Interface qe-1/1/1 type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/2 vlan_mode=trunk tag=1  -- set
Interface qe-1/1/2 type=pica8
```

(3)add a VXLAN port named vxlan1 on qe-1/1/2

```
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1
type=pica8_vxlan options:remote_ip=10.10.10.2 options:local_ip=10.10.10.1
options:vlan=1 options:vnid=1122867 options:udp_dst_port=4789
options:src_mac=C8:0A:A9:04:49:1A  options:dst_mac=C8:0A:A9:9E:14:A5
options:egress_port=qe-1/1/2
```

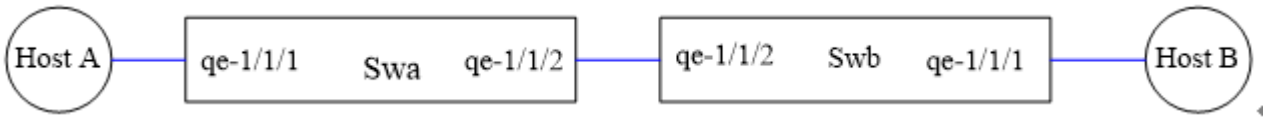
add a flow to switch

```
admin@PicOS-OVS$ovs-ofctl  add-flow br0 in_port=1,actions=output:4097
```

Send packets to qe-1/1/1,qe-1/1/2 will receive pakcets that encapsulated by vxlan header.When vlan of vxlan tunnel is the same with the pvid of qe-1/1/2,the packets from qe-1/1/2 will be stripped vlan of vxlan.Or, packets will have two vlans(outer vlan is vxlan-vlan ,inner vlan is the pvid of ingress port or original vlan of packets)

strip a VXLAN header

topology



configuration

You must configure vxlan port on qe-1/1/2 and qe-1/1/3,and add some flows to the switches so that packets can be encapsulated or decapsulated and forwarded correctly.

(1)create a new bridge named br0.

```
admin@PicOS-OVS$ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
```

(2)add ports to br0.

SwA:

```
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/1 vlan_mode=trunk tag=1  -- set
Interface qe-1/1/1 type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/2 vlan_mode=trunk tag=1  -- set
Interface qe-1/1/2 type=pica8
```

SWb:

```
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/1 vlan_mode=trunk tag=1  -- set
Interface qe-1/1/1 type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/2 vlan_mode=trunk tag=1  -- set
Interface qe-1/1/2 type=pica8
```

(3)add vxlan port vxlan1 on egress port qe-1/1/2 of switcha and switchb

Swa:

```
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1
type=pica8_vxlan options:remote_ip=10.10.10.2 options:local_ip=10.10.10.1
options:vlan=1 options:vnid=1122867 options:udp_dst_port=4789
options:src_mac=C8:0A:A9:04:49:1A  options:dst_mac=C8:0A:A9:9E:14:A5
options:egress_port=qe-1/1/2
```

flow in swa.

```
admin@PicOS-OVS$ovs-ofctl  add-flow br0 in_port=1,actions=output:4097
```

Swb:

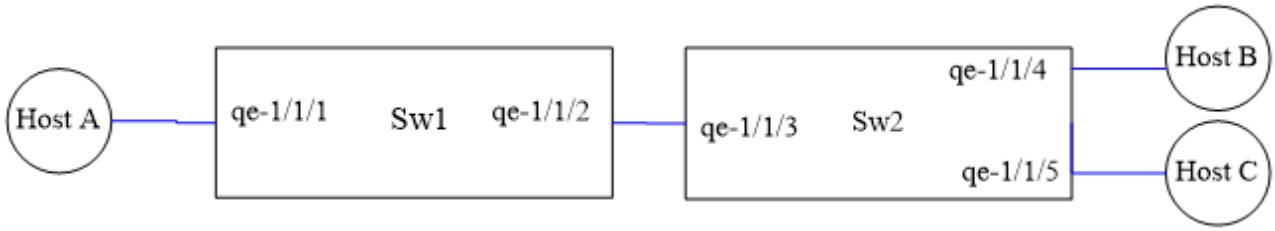
```
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1
type=pica8_vxlan options:remote_ip=10.10.10.1 options:local_ip=10.10.10.2
options:vlan=1 options:vnid=1122867 options:udp_dst_port=4789
options:src_mac= C8:0A:A9:9E:14:A5  options:dst_mac= C8:0A:A9:04:49:1A
options:egress_port=qe-1/1/2
```

```
admin@PicOS-OVS$ovs-ofctl add-flow br0 in_port=4097,actions= output:1
```

send packets to qe-1/1/1 of swa,qe-1/1/1 of switchb will receive the orignal packets(the contents of packets are the same with packets that qe-1/1/1 of swa received).

configure two vxlan tunnels on a pair of physical port

topology



configuration

add two pairs of vxlan ports on qe-1/1/2,qe-1/1/3

sw1:

```
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1
type=pica8_vxlan options:remote_ip=10.10.10.2 options:local_ip=10.10.10.1
options:vlan=1 options:vnid=1122867 options:udp_dst_port=4789
options:src_mac=C8:0A:A9:04:49:1A  options:dst_mac=C8:0A:A9:9E:14:A5
options:egress_port=qe-1/1/2
admin@PicOS-OVS$
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan2 -- set interface vxlan2
type=pica8_vxlan options:remote_ip=10.10.60.1 options:local_ip=10.10.60.2
options:vlan=2 options:vnid=1122869 options:udp_dst_port=4789
options:src_mac=22:22:22:04:49:1A  options:dst_mac=44:44:44:9E:14:A5
options:egress_port=qe-1/1/2
```

flows in sw1,

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,dl_src=22:22:22:22:22:22,actions=output:4097
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,dl_src=22:22:22:22:22:23,actions=output:4098
```

sw2:

```
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1
type=pica8_vxlan options:remote_ip=10.10.10.1 options:local_ip=10.10.10.2
options:vlan=1 options:vnid=1122867 options:udp_dst_port=4789
options:src_mac=C8:0A:A9:9E:14:A5  options:dst_mac=C8:0A:A9:04:49:1A
options:egress_port=qe-1/1/3
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan2 -- set interface vxlan2
type=pica8_vxlan options:remote_ip=10.10.60.2 options:local_ip=10.10.60.1
options:vlan=2 options:vnid=1122869 options:udp_dst_port=4789
options:src_mac=44:44:44:04:49:1A  options:dst_mac=22:22:22:9E:14:A5
options:egress_port=qe-1/1/3
```

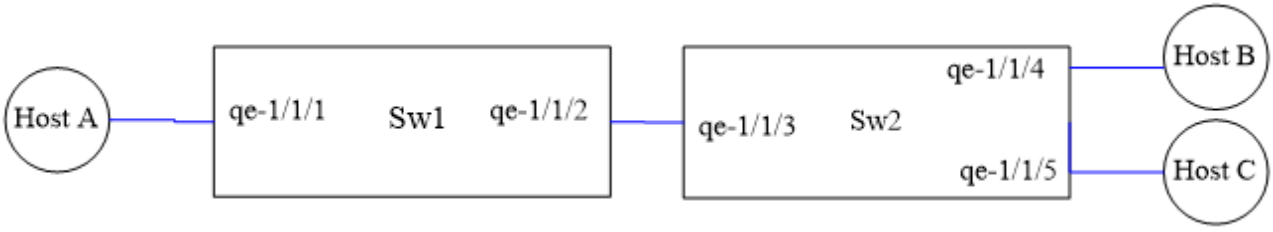
flows in sw2,

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=4097,dl_src=22:22:22:22:22:22,actions=output:4
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=4098,dl_src=22:22:22:22:22:23,actions=output:5
```

send packets to qe-1/1/1 of sw1,qe-1/1/4 should receive the packets with src_mac :22:22:22:22:22:22,and qe-1/1/5 should receive the packets with src_mac 22:22:22:22:22:23.

collaboration between l2gre and vxlan

topology



configuration

You must configure vxlan port and l2gre port on qe-1/1/2 and qe-1/1/3.Add flows on both switches,so packets can be forwarded correctly.

sw1:

```
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1
type=pica8_vxlan options:remote_ip=10.10.10.2 options:local_ip=10.10.10.1
options:vlan=1 options:vnid=1122867 options:udp_dst_port=4789
options:src_mac=C8:0A:A9:04:49:1A  options:dst_mac=C8:0A:A9:9E:14:A5
options:egress_port=qe-1/1/2
admin@PicOS-OVS$
ovs-vsctl add-port br0 l2gre1 -- set Interface l2gre1 type=pica8_l2gre
options:remote_ip=10.10.61.10 options:local_ip=10.10.60.10 options:vlan=1
options:l2gre_key=1234 options:src_mac=C8:0A:A9:22:22:22
options:dst_mac=C8:0A:A9:33:33:33  options:egress_port=qe-1/1/2
```

flows in sw1,

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,dl_dst=22:22:22:22:22:22,actions=output:4097
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,dl_dst=22:22:22:22:22:23,actions=output:5121
```

sw2:

```
admin@PicOS-OVS$ovs-vsctl add-port br0 vxlan1 -- set interface vxlan1
type=pica8_vxlan options:remote_ip=10.10.10.1 options:local_ip=10.10.10.2
options:vlan=1 options:vnid=1122867 options:udp_dst_port=4789
options:src_mac=C8:0A:A9:9E:14:A5 options:dst_mac=C8:0A:A9:04:49:1A
options:egress_port=qe-1/1/3
admin@PicOS-OVS$ovs-vsctl add-port br0 l2gre1 -- set Interface l2gre1
type=pica8_l2gre options:remote_ip=10.10.60.10 options:local_ip=10.10.61.10
options:vlan=1 options:l2gre_key=1234 options:src_mac=C8:0A:A9:33:33:33
options:dst_mac=C8:0A:A9:22:22:22 options:egress_port=qe-1/1/3
```

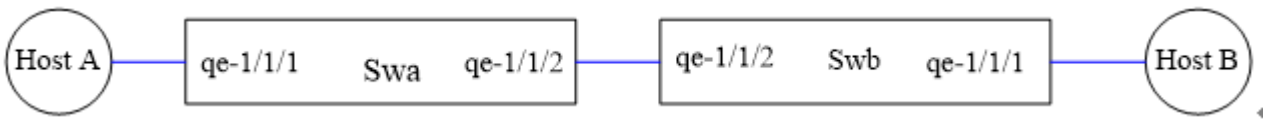
flows in sw2,

```
admin@PicOS-OVS$ovs-ofctl  add-flow br0
in_port=4097,dl_dst=22:22:22:22:22:22,actions=output:4
admin@PicOS-OVS$ovs-ofctl  add-flow br0
in_port=5121,dl_dst=22:22:22:22:22:23,actions=output:5
```

Vnid must be the same when you want to build a vxlan tunnel between two ports.Different vxlan tunnels must have different vnids.Besides,packets are not decapsulated when the vnid is different between the vxlan tunnel.VXLAN can work togther with GRE,L2GRE,VXLAN.

Option:

topology



Ganarally, untag packet from Host A send to Swa will tag pvid in port qe-1/1/1. The new tag packet add VXLAN header and strip VXLAN header through VXLAN tunnel, and will keep the tag forwarding on Swb qe-1/1/1 even though the tag equel the pvid of Swb qe-1/1/1. The result will be Host B receive a tag packet which different with the original packet.

To avoid above issue, pica8 support packet keep untag through pica8 switch port. The following command is necessary.

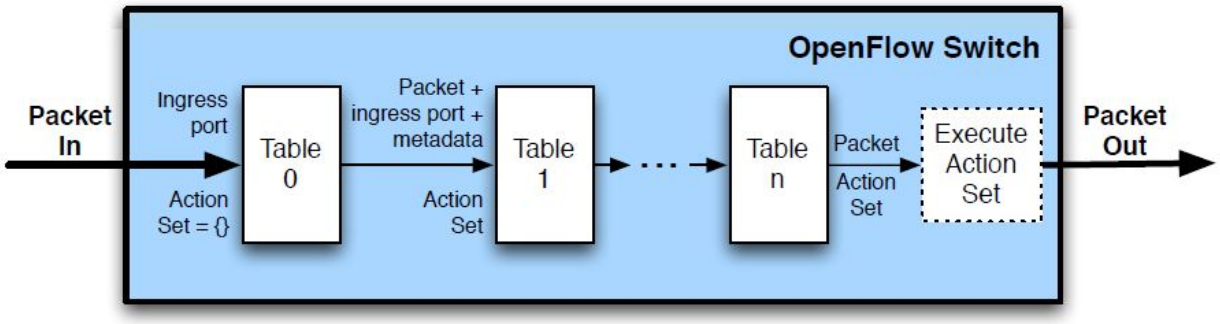
```
ovs-vsctl set interface qe-1/1/1 options:access-vport=true
```

If add the command on Swa, untag packet through in Swa qe-1/1/1 will not tag the pvid, through VXLAN tunnel and keep untag forwarding Swb qe-1/1/1. The result is Host B will receive untag packet.

Configuring Multi-Table

Hardware OpenFlow Multi-table limitations

OpenFlow 1.1 (and later versions) have a concept of tables - independent lookup tables that can be chained in any way you wish. This is a very useful concept to decrease the number of flows in an equipment by segmenting those flows in multiple tables.



The implementation of those multiple tables is not difficult in a software based switch like OVS. But it is a substantial issue for hardware based switch. This is because most ASIC have a limited set of capacities and do multiple lookup on packets is severely limited.

The multi-table concept is very useful though to emulate an ASIC Pipeline. It allows the Openflow based solution to leverage a lot more of the Switch ASIC capacities like complex lookup or different types of memory available on the ASIC. An hardware based multi-table implementation has to be limited though to reflect the limitation of the underlying ASIC.

This means that the number of tables, the conflict between tables, the capacity of those tables and the link between them will be limited by the implementation. This is now defined more generically as a Table Typed Patterned by the ONF.

Multi-Tables in TCAM

Traditionally in an hardware based Openflow implementation, the flows are placed in the switch TCAM memory. This is because this memory is perfect for complex matching (can match on many part of the packet header and the actions possibles once the flow is matched are very diverse) and as such is a good match for most Openflow solution.

In PicOS, by default, all Openflow tables are implemented in TCAM.

It is possible to create multiple tables but because in the TCAM only one table is available, the OS is normalizing the flows into only one hardware table (table 0).

> Because the normalizing process cannot simulate all the types of multi table logic, it is most of the time not recommended to use this TCAM-only Multi-table implementation. It is mainly used as a Proof of concept or demonstration purpose.
> One way to be sure that the normalizing process will render the logic of the flows correctly is to have only one of the tables with actions. All the other tables should only have drop or goto action.

Using the Forwarding database instead of the TCAM

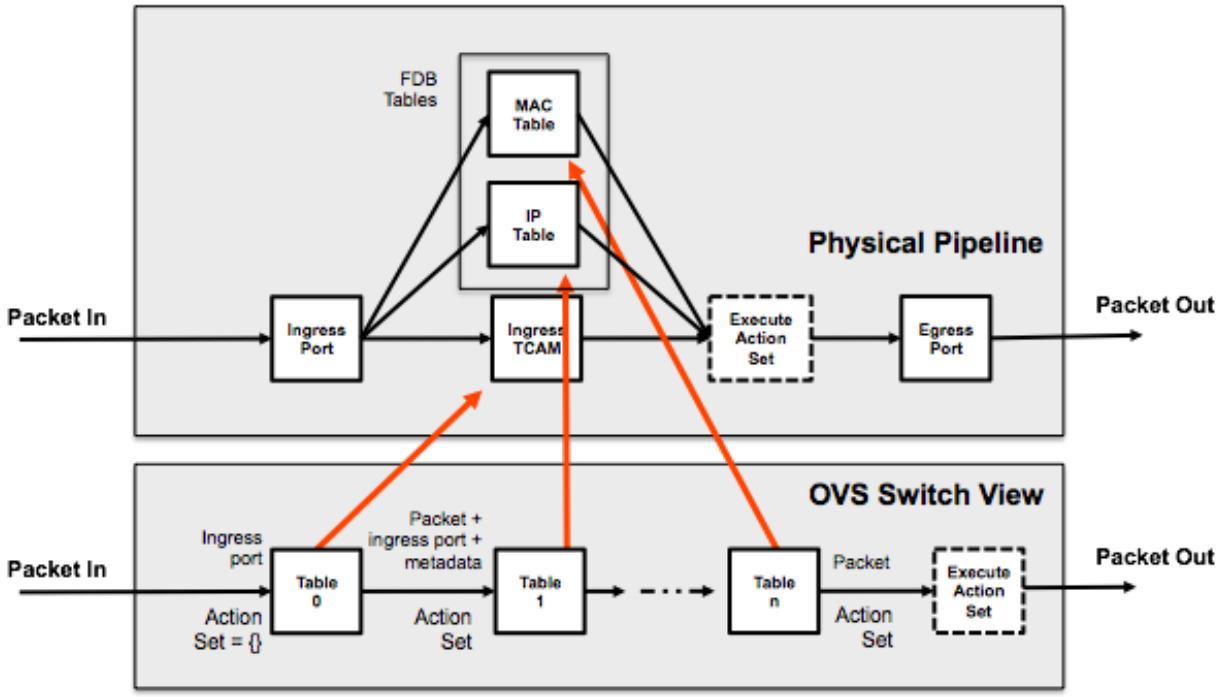
Starting in version 2.4, PicOS supports the FDB (Forwarding database) table or ROUTE table like the traditional L2/L3 mode. That is to say the flows can be stored not only in TCAM table but also in FDB or ROUTE table. See the Switch Hardware Architecture for a description of the actual hardware pipeline.

This is very useful when the scaling of the solution is important and this allows the usage of more memories on the switch as well as access more complex lookup.

The FDB tables consist of a MAC table (similar to a typical L2 Switch Mac lookup) and an IP Table (Similar to a typical L3 Router IP lookup). You can select to download your flows into the TCAM (default) the MAC table or the IP table.

Every packets will match all those tables. Conflict between tables (different action in different tables) is managed by the table priority which can be configured.

> OpenFlow "goto" action is not supported between tables. In this hardware implementation, all tables will be used.



To Map a specific OpenFlow table to the MAC table, use the command:

set-l2-mode TRUE|FALSE [TABLE] command to *enable the MAC table to store flows.* ***[TABLE]*** is the table number which table you set as the FDB table. By default it is the table 251. The flow in the TCAM table should strictly match dl_dst,dl_vlan,(output port in action of flow).

To Map a specific OpenFlow table to the IP table, use the command:

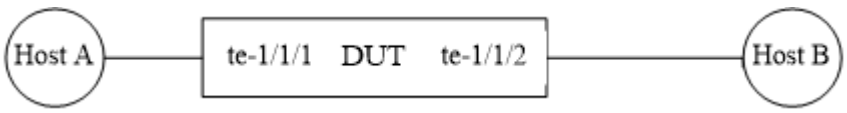
set-l3-mode TRUE|FALSE [TABLE] command to *enable the ROUTE table to store flows.* ***[TABLE]*** is the table number which table you set as the ROUTE table. By default, the ROUTE table number is 252. The flows wanted to be stored in ROUTE must strictly match dl_dst,dl_vlan,dl_type,nw_dst,(mod_dl_dst,output port in action of flow).

By default, TCAM matching has higher priority than L2/L3, and the priority is 0. User can use the command '**ovs-vsctl set-l2-l3-preference true**' to have the FIB/MAC table with a higher priority than the TCAM table.

By default, the ROUTE table is higher priority than the MAC table.

> ⚠ It is possible to have maximum 3 hardware tables with flows in our current implementation simultaneously: 1 TCAM table, 1 ROUTE table and 1 MAC table.

Examples



FDB table configuration example

(1) create a new bridge named br0.

```
admin@PicOS-OVS$ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
```

(2)add ports to br0.

```
admin@PicOS-OVS$ovs-vsctl add-port br0 te-1/1/1 vlan_mode=trunk tag=1  -- set
Interface te-1/1/1 type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 te-1/1/2 vlan_mode=trunk tag=1  -- set
Interface te-1/1/2 type=pica8
```

(3)set l2-mode true without table number

```
admin@PicOS-OVS$ovs-vsctl set-l2-mode true
```

(4)add a flow with table=251

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
table=251,dl_vlan=10,dl_dst=22:22:22:22:22:22,actions=output:2
```

Check the flows in hardware using command ***ovs-appctl pica/dump-flows.*** You will see the flow is stored in l2 table.If you want table 2 to be the FDB table,using ***ovs-vsctl set-l2-mode true 2*** command.

ROUTE table configuration example

(1) create a new bridge named br0.

```
admin@PicOS-OVS$ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
```

(2)add ports to br0.

```
admin@PicOS-OVS$ovs-vsctl add-port br0 te-1/1/1 vlan_mode=trunk tag=1  -- set
Interface te-1/1/1 type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 te-1/1/2 vlan_mode=trunk tag=1  -- set
Interface te-1/1/2 type=pica8
```

(3)set l3-mode true without table number

```
admin@PicOS-OVS$ovs-vsctl set-l3-mode true
```

(4)add a flow with table=252

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
table=252,dl_type=0x0800,dl_dst=22:22:22:22:22:22,nw_dst=192.168.2.30,dl_vlan=1
```

Check the flows in hardware using command ***ovs-appctl pica/dump-flows.*** You will see the flow is stored in l3 table.If you want table 4 to be the FDB table,using ***ovs-vsctl set-l3-mode true 4*** command.

Configuring Network Address Translation

The Network Address Translation (NAT) process maps IP addresses from one address domain (or realm) to another to provide transparent routing to end hosts. Typically, NAT allows organizations to map public external addresses to private or unregistered addresses. P-5101 and P-5401 platforms support this function in ovs mode only.

A flow with NAT actions (changing ip address or L4 port) can be hardware switched. Flows can be associated with the following actions:

mod_nw_dst, mod_nw_src, mod_tp_dst and mod_tp_src

Example 1:the flow that matches dl_dst,and dl_vlan at least can be treated as direct flow.

(1)create a new bridge named br0.

```
admin@PicOS-OVS$ovs-vsctl add-br br0  -- set bridge br0 datapath_type=pica8
```

(2)add ports to br0.

```
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/1 vlan_mode=trunk tag=299 --
set interface qe-1/1/1 type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/2 vlan_mode=trunk tag=299 --
set interface qe-1/1/2 type=pica8
```

(3)add flows to br0.

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,ip,dl_vlan=100,dl_dst=42:22:22:22:22:22,actions=set_field:192.168.5.5
```

(4)check flow tables.

```
admin@PicOS-OVS$ovs-ofctl dump-flows br0
OFPST_FLOW reply (OF1.4) (xid=0x2):
 cookie=0x0, duration=9.480s, table=0, n_packets=n/a, n_bytes=0,
ip,in_port=1,dl_vlan=100,dl_dst=42:22:22:22:22:22
actions=set_field:192.168.5.5->ip_src,output:2
admin@PicOS-OVS$ovs-appctl pica/dump-flows
#1 normal permanent ip,in_port=1,dl_vlan=100,dl_dst=42:22:22:22:22:22,
actions:set(ipv4(src=192.168.5.5,dst=0.0.0.0,proto=0,tos=0,ttl=0,frag=no)),2
Total 1 flows in HW.
admin@PicOS-OVS$
```

Example 2: action with mod_nw_src

(1)create a new bridge named br0

```
admin@PicOS-OVS$ovs-vsctl add-br br0  -- set bridge br0 datapath_type=pica8
```

(2)add ports to br0

```
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/1 vlan_mode=trunk tag=299 --
set interface qe-1/1/1 type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 qe-1/1/2 vlan_mode=trunk tag=299 --
set interface qe-1/1/2 type=pica8
```

(3)add flows to br0.

```
ovs-ofctl add-flow br0
in_port=1,dl_type=0x0800,actions=set_field:192.168.5.5-\>nw_src,output:2
```

(4)Send increasing src_ip packets to qe-1/1/1,qe-1/1/2 will receive the packets with the src_ip 192.168.5.5.

(5)check flow tables

```
admin@PicOS-OVS$ovs-ofctl dump-flows br0
OFPST_FLOW reply (OF1.4) (xid=0x2):
cookie=0x0, duration=1214.855s, table=0, n_packets=17906, n_bytes=6648001232,
ip,in_port=1 actions=set_field:192.168.5.5->ip_src,output:2
admin@PicOS-OVS$ovs-appctl pica/dump-flows
#76 normal
priority=1048560,tcp,in_port=1,nw_src=192.168.2.101,nw_dst=192.168.100.100,nw_t
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
normal
priority=1048560,tcp,in_port=1,nw_src=192.168.2.100,nw_dst=192.168.100.100,nw_t
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
2 flows in HW.
```

From above tables,qe-1/1/2 receive packets with src_ip 192.168.5.5.

Example 3: action with mod_nw_src and mod_tp_src

Establish br0 and add ports in br0 like above configration. And add flow like following:

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
in_port=1,dl_type=0x0800,tcp,actions=set_field:192.168.5.5-\>nw_src,set_field:0
```

Sending increasing dst_mac packets to qe-1/1/1, mac address from 22:22:22:22:22:22 to 22:22:22:22:22:2b, then check tables:

```
admin@PicOS-OVS$ovs-ofctl dump-flows br0
OFPST_FLOW reply (OF1.4) (xid=0x2):
cookie=0x0, duration=48.298s, table=0, n_packets=n/a, n_bytes=30526416,
tcp,in_port=1
actions=set_field:192.168.5.5->ip_src,set_field:1110->tcp_src,output:4
admin@PicOS-OVS$
admin@PicOS-OVS$ovs-appctl pica/dump-flows
#103 normal
priority=1048560,tcp,in_port=1,dl_vlan=2000,dl_dst=22:22:22:22:22:29,nw_src=192
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
normal
priority=1048560,tcp,in_port=1,dl_vlan=2000,dl_dst=22:22:22:22:22:23,nw_src=192
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
normal
priority=1048560,tcp,in_port=1,dl_vlan=2000,dl_dst=22:22:22:22:22:27,nw_src=192
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
normal
priority=1048560,tcp,in_port=1,dl_vlan=2000,dl_dst=22:22:22:22:22:26,nw_src=192
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
normal
priority=1048560,tcp,in_port=1,dl_vlan=2000,dl_dst=22:22:22:22:22:24,nw_src=192
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
normal
priority=1048560,tcp,in_port=1,dl_vlan=2000,dl_dst=22:22:22:22:22:2b,nw_src=192
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
normal
priority=1048560,tcp,in_port=1,dl_vlan=2000,dl_dst=22:22:22:22:22:2a,nw_src=192
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
normal
priority=1048560,tcp,in_port=1,dl_vlan=2000,dl_dst=22:22:22:22:22:22,nw_src=192
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
normal
priority=1048560,tcp,in_port=1,dl_vlan=2000,dl_dst=22:22:22:22:22:28,nw_src=192
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
normal
priority=1048560,tcp,in_port=1,dl_vlan=2000,dl_dst=22:22:22:22:22:25,nw_src=192
actions:set(ipv4(src=192.168.5.5,dst=192.168.100.100,proto=6,tos=0,ttl=64,frag=
10 flows in HW.
admin@PicOS-OVS$
```

And qe-1/1/2 receive packets with the src_ip 192.168.5.5 and src_port 1110.

Due to ASIC limitation, a flow can not modify l4_src_port without modifying SIP or modify l4_dst_port without modifying DIP.

If only modifying SIP(DIP) or SIP+L4_SRC_PORT(DIP+L4_DST_PORT), up to 2k flow can be configured.If modifying both SIP[|L4_SRC_PORT] and DIP[|L4_DST_PORT],the number of flow supported are 1k.

ASIC Limitation

Because some limitation of ASIC, some flow installed hardware can not work as expected. Please refer these chap before you start to trouble-shoot the issues.

udp/ip, tcp/ip

when you add some flows with the same priority,and one flow's match fields includes another flow's match fields,then the action of flow is at random.For example:

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
priority=10000,ip,in_port=14,dl_vlan=2,actions=push_vlan:0x8100,set_field:2503-
pica/dump-flows
#40 permanent priority=10000,ip,in_port=14,dl_vlan=2,
actions:push_vlan(vid=2503),mod_vlan_pcp(pcp=0),15
Total 1 flows in TCAM.
admin@PicOS-OVS$ovs-ofctl add-flow br0
priority=10000,udp,in_port=14,dl_vlan=2,tp_dst=2123,actions=push_vlan:0x8100,se

admin@PicOS-OVS$
admin@PicOS-OVS$ovs-appctl pica/dump-flows
#41 permanent priority=10000,udp,in_port=14,dl_vlan=2,tp_dst=2123,
actions:push_vlan(vid=2503),mod_vlan_pcp(pcp=0),15
#40 permanent priority=10000,ip,in_port=14,dl_vlan=2,
actions:push_vlan(vid=2503),mod_vlan_pcp(pcp=0),15
Total 2 flows in TCAM.
```

If you don't want this result,please modify the two flows' priority.

```
admin@PicOS-OVS$ovs-ofctl add-flow br0
priority=12000,udp,in_port=14,dl_vlan=2,tp_dst=2123,actions=push_vlan:0x8100,se
add-flow br0
priority=10000,ip,in_port=14,dl_vlan=2,actions=push_vlan:0x8100,set_field:2503-
pica/dump-flows
#42 permanent priority=12000,udp,in_port=14,dl_vlan=2,tp_dst=2123,
actions:push_vlan(vid=2500),mod_vlan_pcp(pcp=0),15
#40 permanent priority=10000,ip,in_port=14,dl_vlan=2,
actions:push_vlan(vid=2503),mod_vlan_pcp(pcp=0),15
Total 2 flows in TCAM.
```

Configuring CFM

Connectivity Fault Management(CFM) is an IEEE standard, 802.1g which specifies protocols, procedures, and managed objects to support transport fault management. CFM is used for detecting link connectivity fault, confirming the fault and locating the fault occurred in the network.

1. Monitor connectivity to a remote maintenance point on ge-1/1/1:

Set the MPID of CFM:

```
admin@PicOS-OVS$ ovsvsctl set Interface ge-1/1/1 cfm_mpid=2333
```

A Maintenance Point ID (MPID) uniquely identifies each endpoint within a Maintenance Association. According to the 802.1ag specification, MPIDs can only range between [1, 8191].

Set extended mode:

```
admin@PicOS-OVS$ovsvsctl set Interface ge-1/1/1
other_config:cfm_extended=true
```

Extended mode increases the accuracy of the cfm_interval configuration parameter by breaking wire compatibility with 802.1ag compliant implementations. An extended mode allows **eight byte MPIDs**.

Set demand mode:

```
admin@PicOS-OVS$ovsvsctl set Interface ge-1/1/1 other_config:cfm_demand=true
```

When true, and cfm_extended is true, the CFM module operates in demand mode. By default it is set to false.When in demand mode, traffic received on the Interface is used to indicate liveness. CCMs are still transmitted and received.At least one CCM must be received every 100 * cfm_interval amount of time. Otherwise,even if traffic is received, the CFM module will trigger the connectivity fault.Demand mode disables itself when there are multiple remote maintenance points.

Set the requested transmission interval:

```
admin@PicOS-OVS$ovsvsctl set Interface ge-1/1/1
other_config:cfm_interval=10000
```

In standard mode supports intervals of 3, 10, 100, 1000, 10000,60000, or 600000 ms are supported.Extended mode supports any interval up to 65535 ms and default is 1000 ms. However, we do not recommend intervals less than 100 ms.

Set CCM Vlan tag:

```
admin@PicOS-OVS$ovsvsctl set Interface ge-1/1/1
other_config:cfm_ccm_vlan=2000
admin@PicOS-OVS$ovsvsctl set Interface ge-1/1/1
other_config:cfm_ccm_vlan=random
```

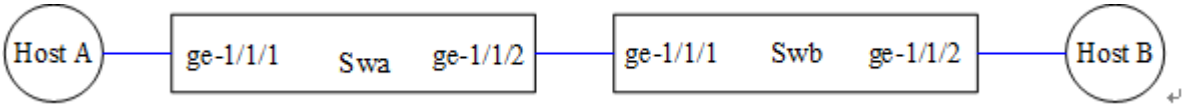
When set, the CFM module will apply a VLAN tag to all CCMs it generates with the given value.

Set CCM Priority:

```
admin@PicOS-OVS$ovsvsctl set Interface ge-1/1/1 other_config:cfm_ccm_pcp=7
```

When set, the CFM module will apply a VLAN tag to all CCMs it generates with the given PCP value, the VLAN ID of the tag is governed by the value of "cfm_ccm_vlan". If "cfm_ccm_vlan" is unset, a VLAN ID of zero is used.

2. CFM Example:



Step1:basic configure

DUT1:

```
admin@PicOS-OVS$ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 ge-1/1/1 vlan_mode=trunk tag=1  -- set
Interface ge-1/1/1 type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 ge-1/1/2 vlan_mode=trunk tag=1  -- set
Interface ge-1/1/2 type=pica8
admin@PicOS-OVS$ovs-vsctl -- set bridge br0 mirrors=@m -- --id=@ge-1/1/2 get
Port ge-1/1/2 -- --id=@ge-1/1/1 get Port ge-1/1/1 -- --id=@m create Mirror
name=mymirror select-src-port=@ge-1/1/2 output-port=@ge-1/1/1
```

DUT2:

```
admin@PicOS-OVS$ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 ge-1/1/1 vlan_mode=trunk tag=1  -- set
Interface ge-1/1/1 type=pica8
admin@PicOS-OVS$ovs-vsctl add-port br0 ge-1/1/2 vlan_mode=trunk tag=1  -- set
Interface ge-1/1/2 type=pica8
admin@PicOS-OVS$ovs-vsctl -- set bridge br0 mirrors=@m -- --id=@ge-1/1/1 get
Port ge-1/1/1 -- --id=@ge-1/1/2 get Port ge-1/1/2 -- --id=@m create Mirror
name=mymirror select-src-port=@ge-1/1/1 output-port=@ge-1/1/2
```

Step2:configure cfm:

DUT1:

```
admin@PicOS-OVS$ovs-vsctl set interface ge-1/1/2 cfm-mpid=8999
admin@PicOS-OVS$ovs-vsctl set interface ge-1/1/2
other_config:cfm_extended=true
```

DUT2:

```
admin@PicOS-OVS$ovs-vsctl set interface ge-1/1/1 cfm-mpid=9000
admin@PicOS-OVS$ovs-vsctl set interface ge-1/1/1
other_config:cfm_extended=true
```

Step3:check packets

DUT1:

Check list interface:

```
admin@PicOS-OVS$ovs-vsctl list interface ge-1/1/2
_uuid : 94942d57-d9a8-4030-ad3b-483dadbd7926
admin_state : up
bfd : {}
bfd_status : {}
cfm_fault : false
cfm_fault_status : []
cfm_flap_count : 2
cfm_health : []
cfm_mpid : 8999
cfm_remote_mpids : [9000]
cfm_remote_opstate : up
duplex : full
external_ids : {}
ifindex : 13
ingress_policing_burst: 0
ingress_policing_rate: 0
lacp_current : []
link_resets : 0
link_speed : 1000000000
link_state : up
mac : []
mac_in_use : "00:e0:ec:25:2d:5e"
mtu : 9212
name : "ge-1/1/2"
ofport : 13
ofport_request : []
options : {}
other_config : {cfm_extended="true"}
statistics : {collisions=0, rx_bytes=3255, rx_crc_err=0, rx_dropped=28,
rx_errors=0, rx_frame_err=0, rx_over_err=0, rx_packets=35, tx_bytes=1395,
tx_dropped=0, tx_errors=0, tx_packets=15}
status : {}
type : "pica8"
wred_queues : {}
```

Check cfm/show:

```
admin@PicOS-OVS$ovs-appctl  cfm/show
---- ge-1/1/2 ----
MPID 8999: extended
                average health: undefined
                opstate: up
                remote_opstate: up
                interval: 1000ms
                next CCM tx: 481ms
                next fault check: 973ms
Remote MPID 9000
                recv since check: true
                opstate: up
admin@PicOS-OVS$
```

Check hardware table:

```
admin@PicOS-OVS$ovs-appctl pica/dump-flows
#168 normal permanent
priority=18000000,in_port=2,dl_dst=01:23:20:00:00:30,dl_type=0x8902,
actions:userspace(pid=0,slow_path(cfm))
#167 normal permanent priority=0, actions:drop
Total 2 flows in HW.
```

DUT2:

Check list interface:

```
admin@PicOS-OVS$ovs-vsctl list interface ge-1/1/1
_uuid : 61bb8ef5-30f9-4855-8cfa-f1ee0bc5b154
admin_state : up
bfd : {}
bfd_status : {}
cfm_fault : false
cfm_fault_status : []
cfm_flap_count : 0
cfm_health : []
cfm_mpid : 9000
cfm_remote_mpids : [8999]
cfm_remote_opstate : up
duplex : full
external_ids : {}
ifindex : 11
ingress_policing_burst: 0
ingress_policing_rate: 0
lacp_current : []
link_resets : 0
link_speed : 1000000000
link_state : up
mac : []
mac_in_use : "08:9e:01:a8:00:49"
mtu : 9212
name : "ge-1/1/11"
ofport : 11
ofport_request : []
options : {}
other_config : {cfm_extended="true"}
statistics : {collisions=0, rx_bytes=1302, rx_crc_err=0, rx_dropped=8,
rx_errors=0, rx_frame_err=0, rx_over_err=0, rx_packets=14, tx_bytes=558,
tx_dropped=0, tx_errors=0, tx_packets=6}
status : {}
type : "pica8"
wred_queues : {}
admin@PicOS-OVS$
```

Check cfm/show:

```
admin@PicOS-OVS$ovs-appctl  cfm/show
---- ge-1/1/1 ----
MPID 9000: extended
                average health: undefined
                opstate: up
                remote_opstate: up
                interval : 1000ms
                next CCM tx: 802ms
                next fault check: 1254ms
Remote MPID 8999
                recv since check: true
                opstate: up
admin@PicOS-OVS$
```

Check hardware table:

```
admin@PicOS-OVS$ovs-appctl pica/dump-flows
#168 normal permanent
priority=18000000,in_port=1,dl_dst=01:23:20:00:00:30,dl_type=0x8902,
actions:userspace(pid=0,slow_path(cfm))
#167 normal permanent priority=0, actions:drop
Total 2 flows in HW.
```

Note:

Standard mode, dl_mac is 01:80:c2:00:00:30; when extended mode,the dst mac is 01:23:20:00:00:30.

OVS Configuration File

The OVS configuration is stored in **/ovs/ovs-vswitchd.conf.db**.

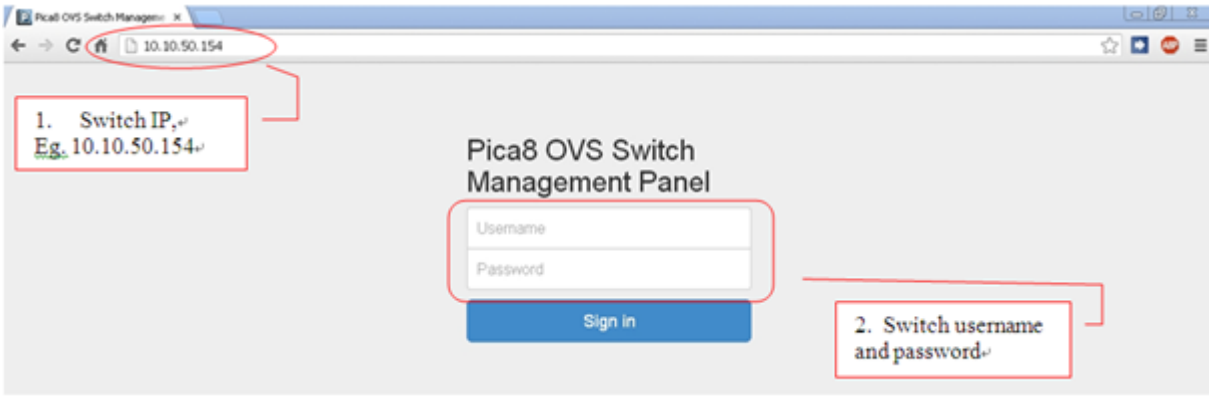
```
admin@PicOS-OVS$cd /ovs/
admin@PicOS-OVS$ls
bin  etc  lib  ovs-vswitchd.conf.db  sbin  share  var
admin@PicOS-OVS$
admin@PicOS-OVS$
```

OVS Web User Interface

- Login Interface
- Monitor
- Adding a Bridge
- Add a Port
- Add GRE Port
- Add Group Table
- Add or Edit a Controller
- Edit Flow Tables
- Edit Lag Interface

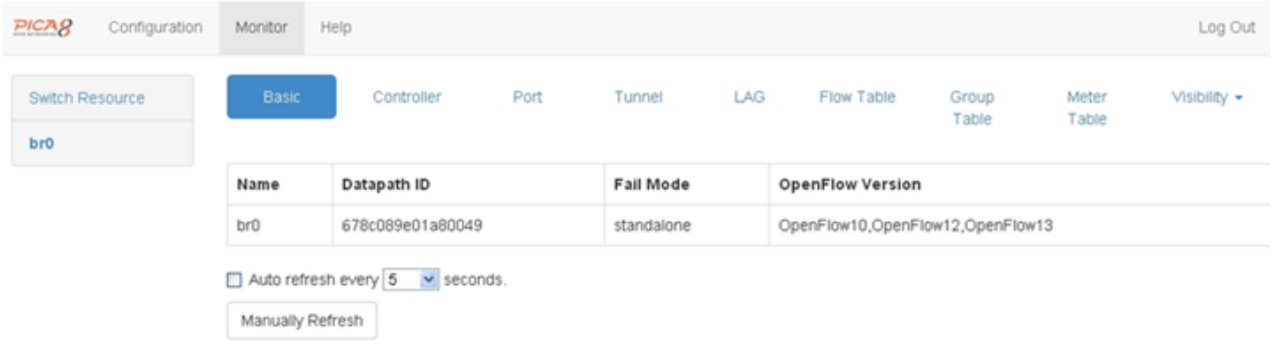
Login Interface

If the switch is running PicOS Version 2.2, enter the switch IP address to launch OVS Web User Interface.



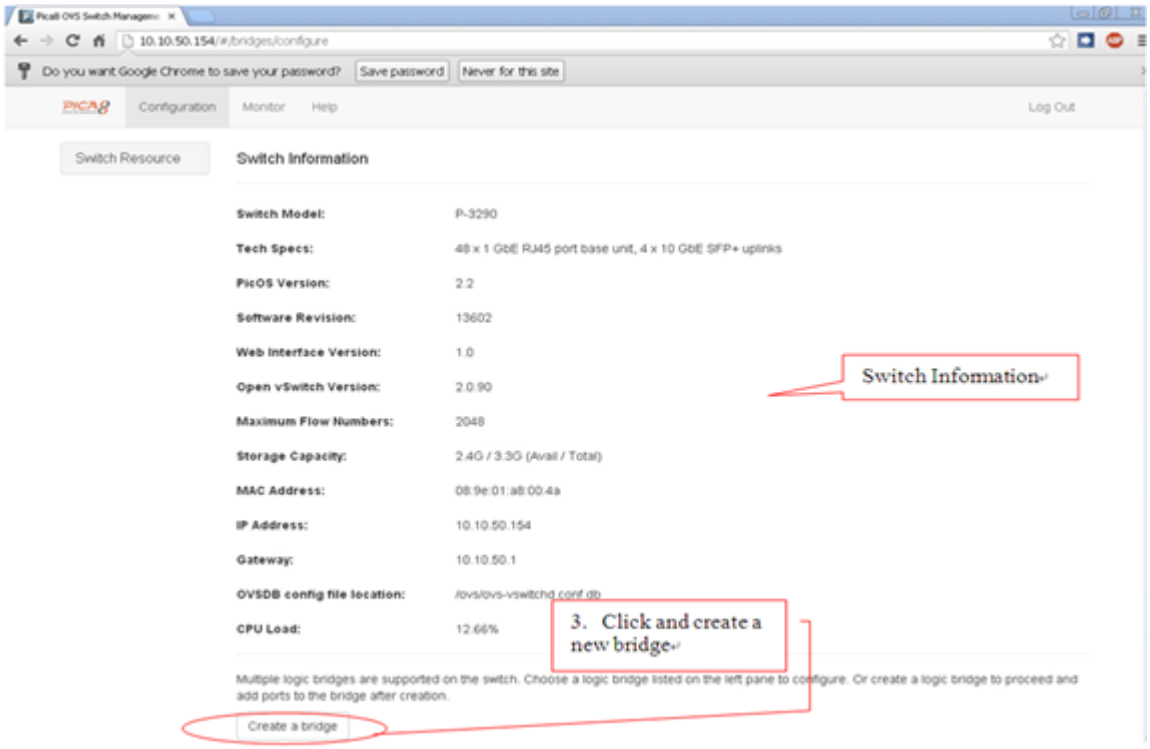
Monitor

The Monitor tab allows you to check information on the switch. You can also adjust the Auto refresh or manually refresh from the monitor tab view.

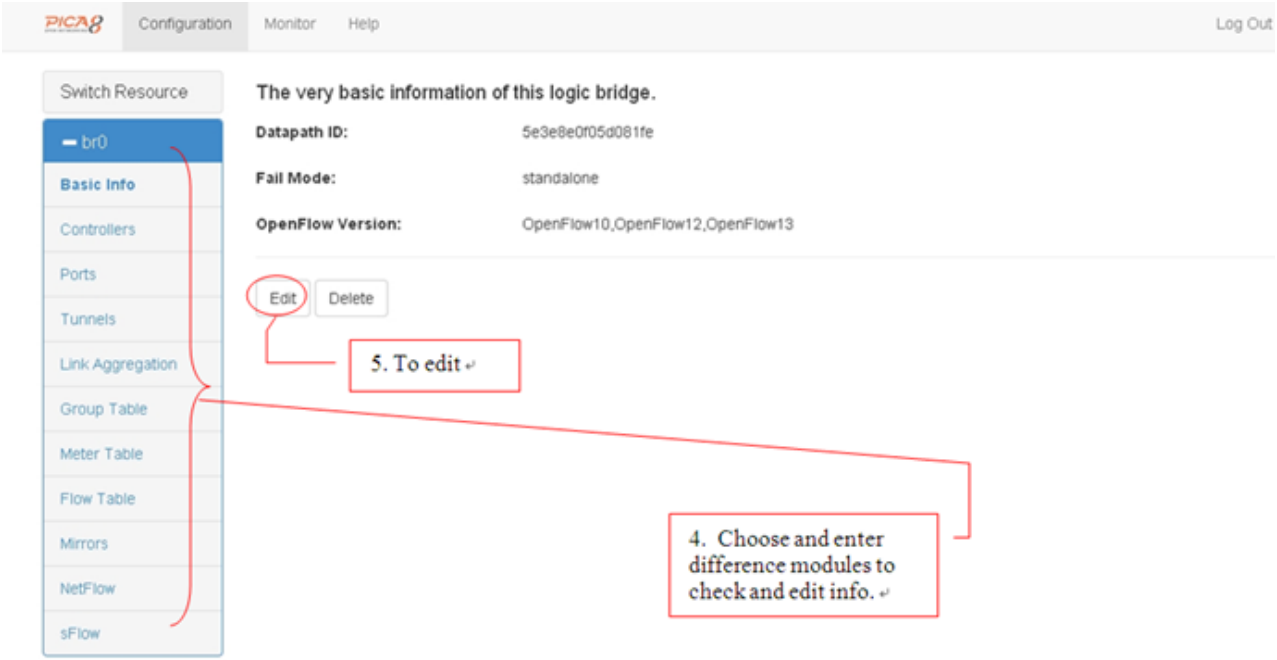


Adding a Bridge

Once you have successfully launched the user interface, the Configuration tab reveals the Switch Resource section that provides basic switch information. To create a bridge, click on the create a new bridge icon.

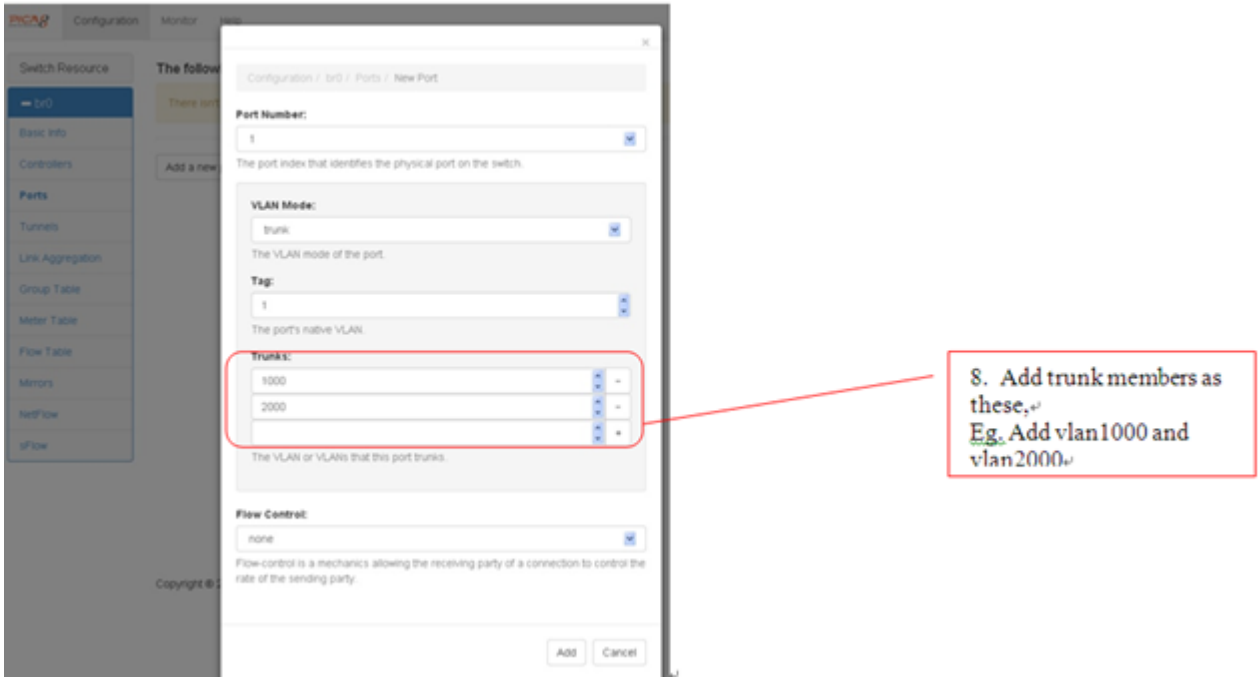


Once you have created a new bridge (in the example below br0), you can delete the bridge or edit the bridge's properties. The menu on the left (in the graphic below) allows you to view, edit and change any of the modules listed in the menu.



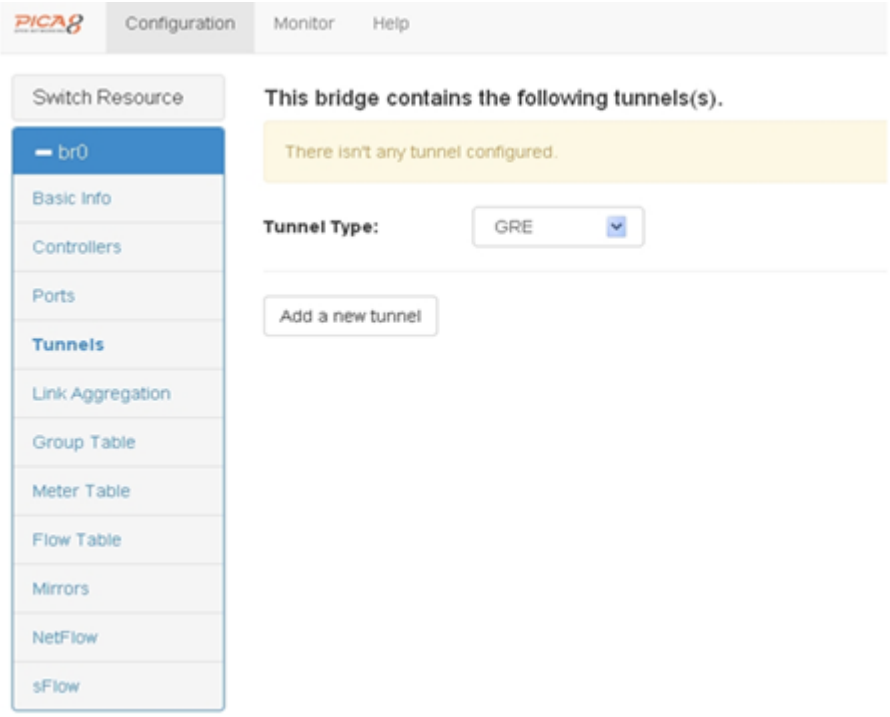
Add a Port

Click on Ports to add a new port. Fill in the port number, VLAN mode, Tag, and Trunks and click Add.

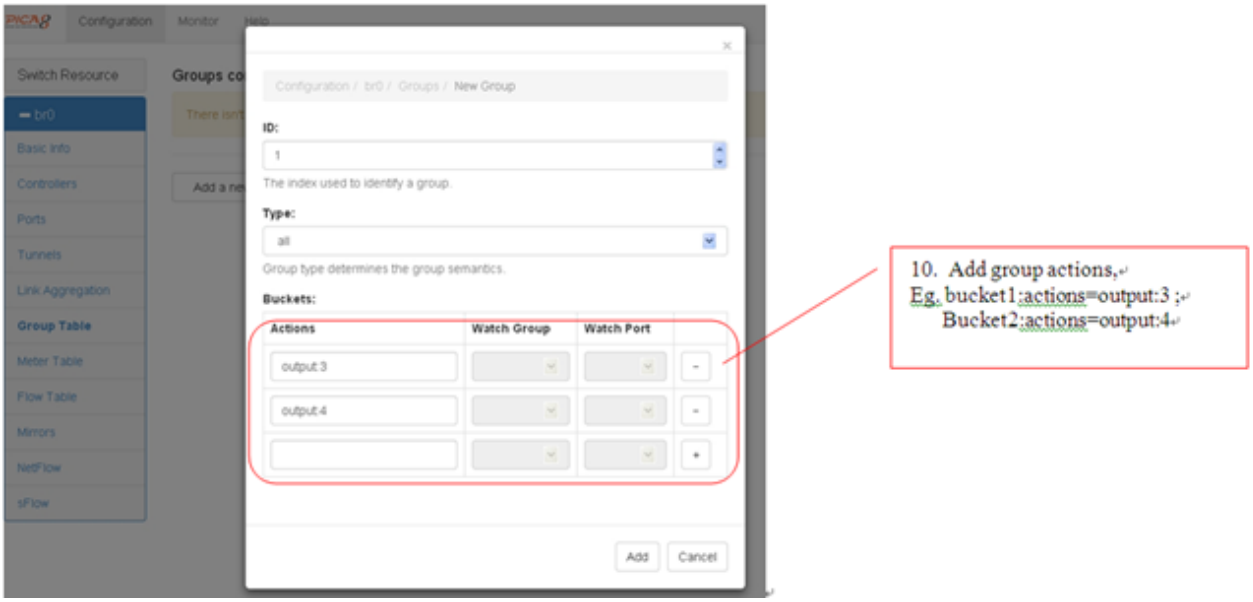


Add GRE Port

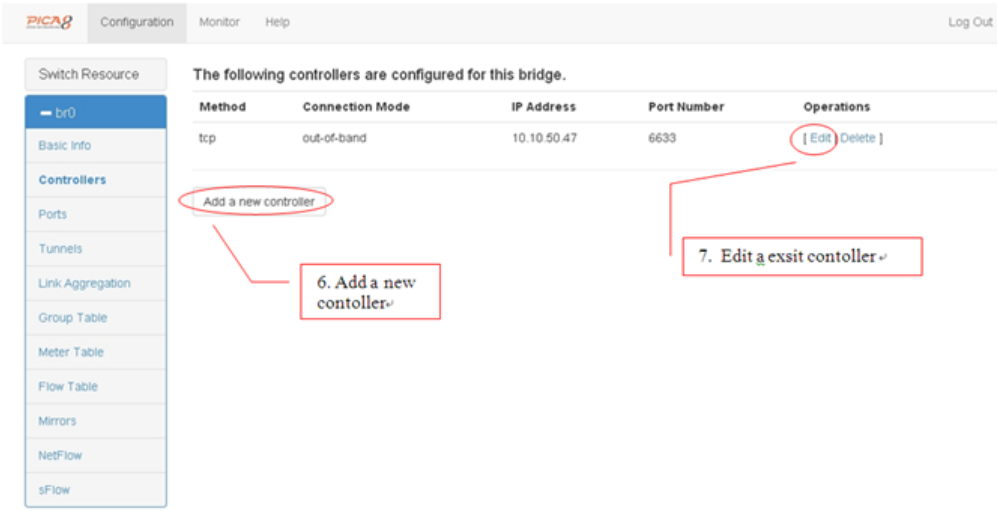
Select Tunnels from the menu to view the bridge's tunnel type or to add or edit a tunnel.



Add Group Table

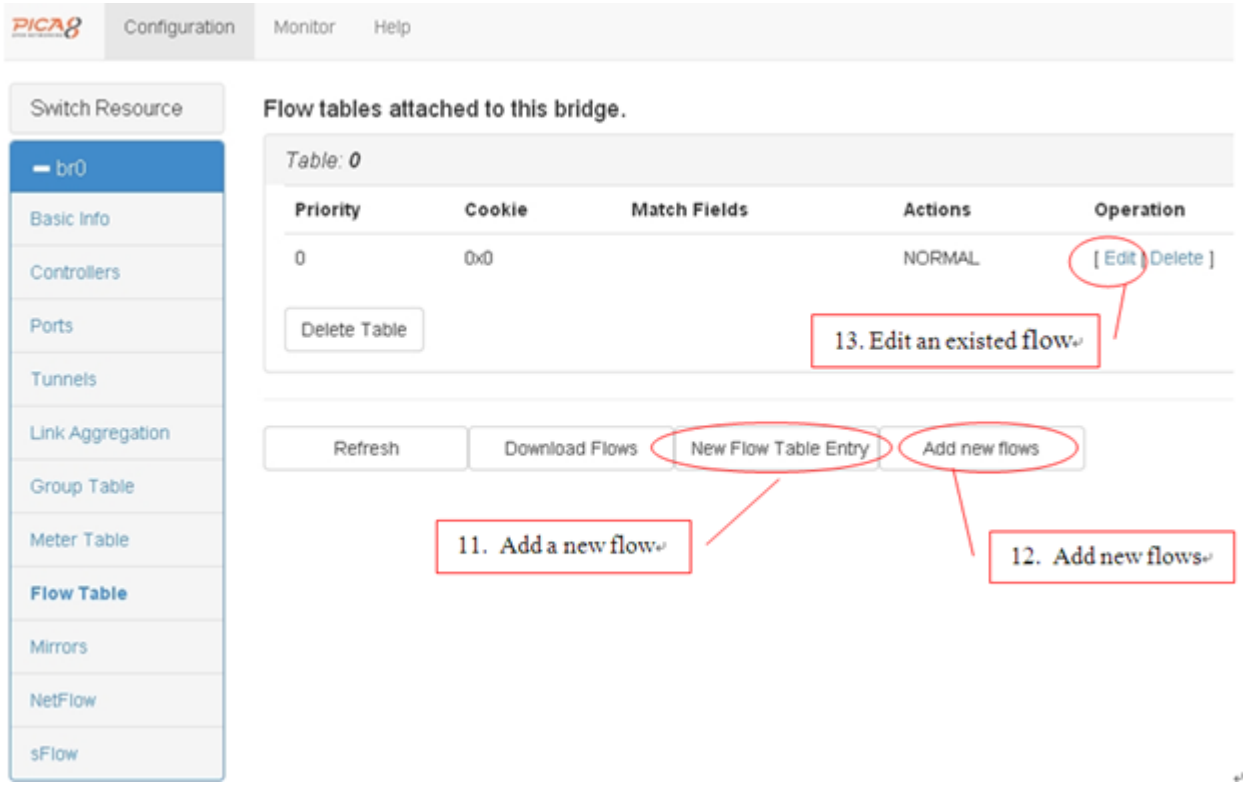


Add or Edit a Controller

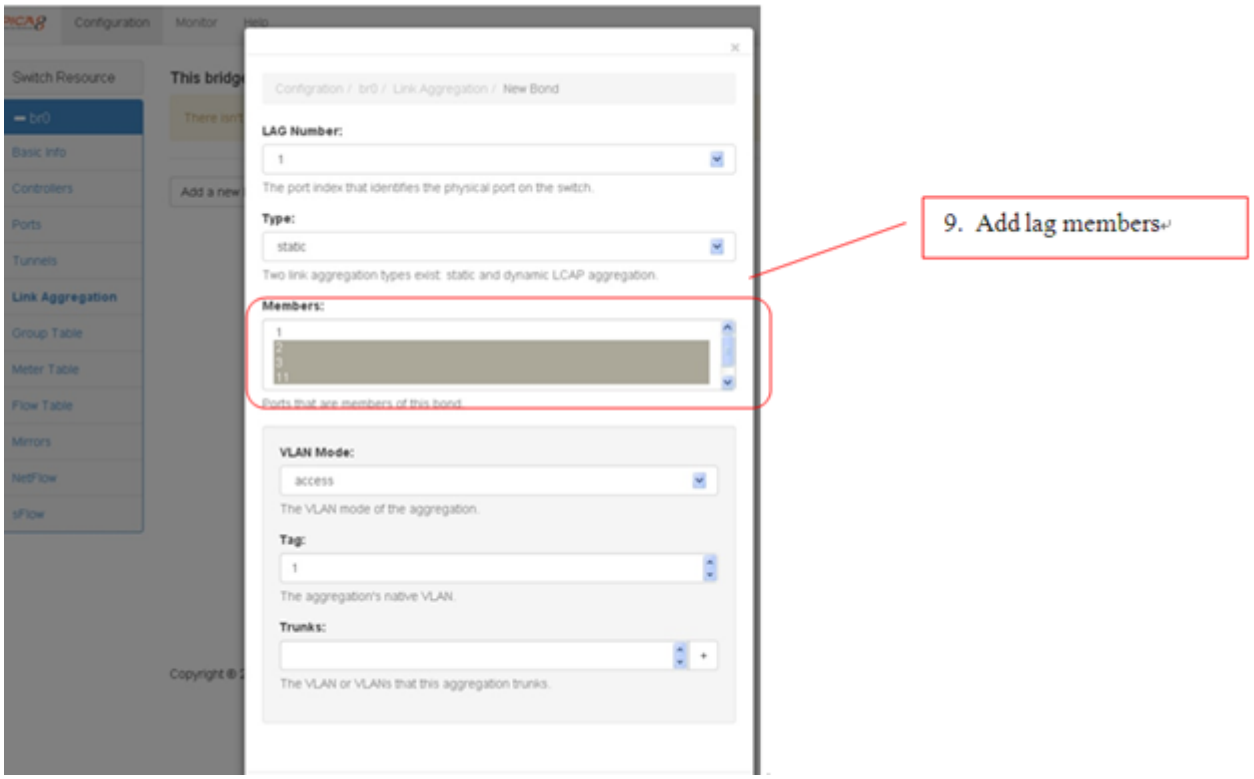


Edit Flow Tables

You can view the flow table attached to the bridge and delete, edit, download, and add to the flow table.



Edit Lag Interface



Examples and Topologies

This chapter gives some configuration example for 802.1Q.

- 802.1Q VLAN
- ECMP
- GRE Tunnel
- MPLS Network
- Multiple Virtual Bridges
- SSL Connection to Controller

802.1Q VLAN

In following topology, we need configure 2 VLANs in switch A and B.

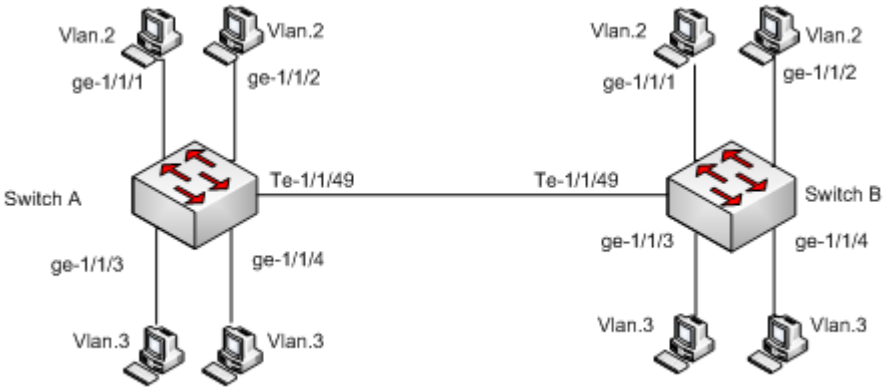


Figure 4-1. 802.1Q network configuration

(1) Configure Switch-A

In switch-A, you need configure ge-1/1/1~ ge-1/1/4 as access port while te-1/1/49 as trunk port, because the 10Gbit link will trunk the traffic of VLAN-2 and VLAN-3

```
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/1 vlan_mode=access tag=2 -- set
Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl  add-port br0 te-1/1/2 vlan_mode=access tag=2 --
set Interface te-1/1/2 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl  add-port br0 te-1/1/3 vlan_mode=access tag=3 --
set Interface te-1/1/3 type=pica8
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/4 vlan_mode=access tag=3 -- set
Interface te-1/1/4 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl  add-port br0 te-1/1/49 vlan_mode=trunk trunk=2,3
-- set Interface te-1/1/49 type=pica8
root@PicOS-OVS#
```

(2) Configure Switch-B

In switch-B, you need configure ge-1/1/1~ ge-1/1/4 as access port while te-1/1/49 as trunk port, because the 10Gbit link will trunk the traffic of VLAN-2 and VLAN-3

```
root@PicOS-OVS# ovs-vsctl  add-port br0 te-1/1/1 vlan_mode=access tag=2 --
set Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl  add-port br0 te-1/1/2 vlan_mode=access tag=2 --
set Interface te-1/1/2 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl  add-port br0 te-1/1/3 vlan_mode=access tag=3 --
set Interface te-1/1/3 type=pica8
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/4 vlan_mode=access tag=3 -- set
Interface te-1/1/4 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl  add-port br0 te-1/1/49 vlan_mode=trunk trunk=2,3
-- set Interface te-1/1/49 type=pica8
root@PicOS-OVS#
```

ECMP

```
root@PicOS-OVS#ovs-vsctl  del-br br0
root@PicOS-OVS#ovs-vsctl  add-br br0 -- set bridge br0 datapath_type=pica8
root@PicOS-OVS#ovs-vsctl  add-port br0 ge-1/1/1 vlan_mode=trunk tag=1
trunks=1000,2000,3000,4094 -- set Interface ge-1/1/1 type=pica8
root@PicOS-OVS#ovs-vsctl  add-port br0 ge-1/1/2 vlan_mode=trunk tag=1
trunks=1000,2000,3000,4094 -- set Interface ge-1/1/2 type=pica8
root@PicOS-OVS#ovs-vsctl  add-port br0 ge-1/1/3 vlan_mode=trunk tag=1
trunks=1000,2000,3000,4094 -- set Interface ge-1/1/3 type=pica8
root@PicOS-OVS#ovs-vsctl  add-port br0 ge-1/1/4 vlan_mode=trunk tag=1
trunks=1000,2000,3000,4094 -- set Interface ge-1/1/4 type=pica8
root@PicOS-OVS#ovs-ofctl del-flows br0
root@PicOS-OVS#ovs-ofctl add-group br0
group_id=1,type=select,bucket=output:2,bucket=output:3,bucket=output:4
root@PicOS-OVS#ovs-ofctl add-flow br0
dl_type=0x0800,nw_dst=192.168.2.0/24,actions=group:1
```

send packets (nw_dst incr number is 200)to port 1,

packets whose nw_dst= 192.168.2.0/255.255.255.3 will forward to port 2.

packets whose nw_dst= 192.168.2.1/255.255.255.3 will forward to port 3.

packets whose nw_dst= 192.168.2.2/255.255.255.3 will forward to port 4.

packets whose nw_dst= 192.168.2.3/255.255.255.3 will forward to port 2.

GRE Tunnel

In following topology, we need configure a GRE tunnel between switch A and B. The IP address of the GRE tunnel is 10.10.61.10/24 and 10.10.60.10/24.

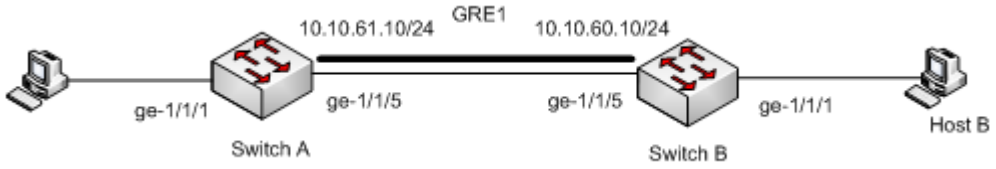


Figure 4-2. GRE tunnel configuration

Configure Switch-A

In switch-A, you need configure a GRE tunnel and two flows as following:

```
root@PicOS-OVS# ovs-vsctl  add-br br0 -- set bridge br0 datapath_type=pica8
root@PicOS-OVS# ovs-vsctl  add-port br0 ge-1/1/1 vlan_mode=trunk tag=1 -- set
Interface ge-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl  add-port br0 ge-1/1/5 vlan_mode=trunk tag=1 -- set
Interface ge-1/1/5 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl add-port br0 gre1 -- set Interface gre1
type=pica8_gre options:remote_ip=10.10.60.10 options:local_ip=10.10.61.10
options:vlan=1 options:src_mac=00:11:11:11:11:11
options:dst_mac=00:22:22:22:22:22 options:egress_port=ge-1/1/5
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:109
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=5,actions=mod_dl_src:00:11:11:11:11:11,mod_dl_dst:00:33:33:33:33:33,out
```

Configure Switch-B

In switch-A, you also need configure a GRE tunnel and two flows as following:

```
root@PicOS-OVS# ovs-vsctl  add-br br0 -- set bridge br0 datapath_type=pica8
root@PicOS-OVS# ovs-vsctl  add-port br0 ge-1/1/1 vlan_mode=trunk tag=1 -- set
Interface ge-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl  add-port br0 ge-1/1/5 vlan_mode=trunk tag=1 -- set
Interface ge-1/1/5 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl  add-port br0 gre1 -- set Interface gre1
type=pica8_gre options:remote_ip=10.10.61.10 options:local_ip=10.10.60.10
options:vlan=1 options:src_mac=00:22:22:22:22:22
options:dst_mac=00:11:11:11:11:11 options:egress_port=ge-1/1/5
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0 in_port=1,actions=output:91
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=5,actions=mod_dl_src:00:22:22:22:22:22,mod_dl_dst:00:66:66:66:66:66,out
```

MPLS Network

In following topology, we configure a simple MPLS network. Traffic (Red) from host-A to host-B will forward by MPLS network with Label 10. The traffic (Blue) from host-C to host-D will forward by MPLS network with Label 20. All the flow will only push ONE MPLS header.

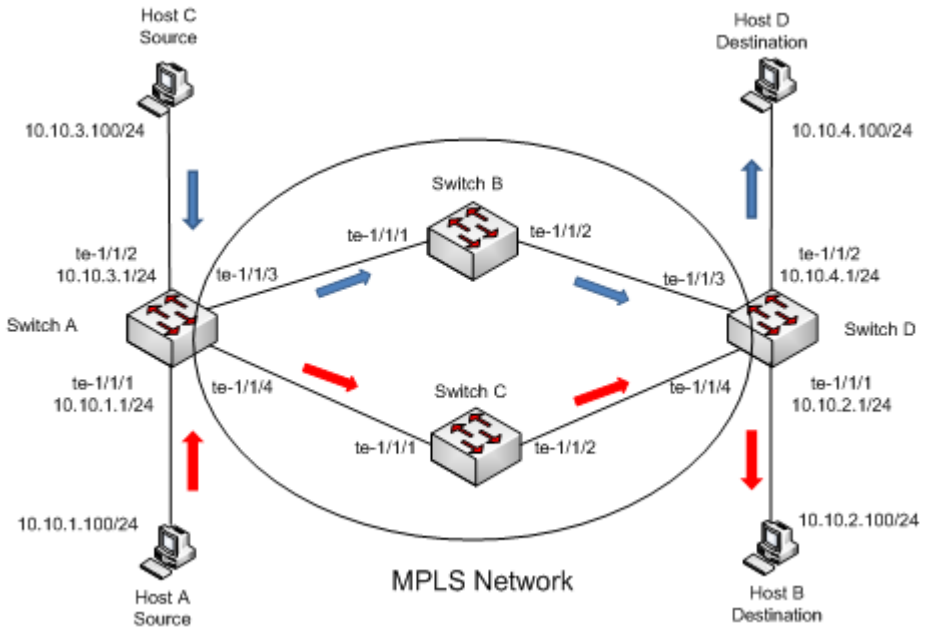


Figure 4-2. MPLS network configuration

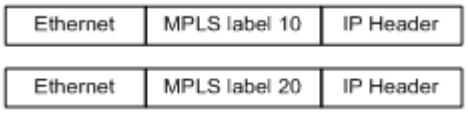
(1) Configure Switch-A

In switch-A, you need configure two flow which will push the MPLS Label 10 and 20 for traffic RED and BLUE respectively.

```
root@PicOS-OVS# ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
device br0 entered promiscuous mode
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/1 vlan_mode=access tag=1 -- set
Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/2 vlan_mode=access tag=1 -- set
Interface te-1/1/2 type=pica8
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/3 vlan_mode=access tag=1 -- set
Interface te-1/1/3 type=pica8
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/4 vlan_mode=access tag=1 -- set
Interface te-1/1/4 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_type=0x0800,nw_src=10.10.1.100,nw_dst=10.10.2.100,dl_vlan=1,action
ovs-ofctl add-flow br0
in_port=2,dl_type=0x0800,nw_src=10.10.3.100,nw_dst=10.10.4.100,dl_vlan=1,action
```

The received packet format in port te-1/1/1 and te-1/1/2 is shown as following (ingress):

The transmitted packet format to port te-1/1/3 and te-1/1/4 is shown as following (egress):



(2) Configure Switch-B

In switch-B, you need configure one flow which will SWAP the MPLS Label 20 to 200 for traffic BLUE.

```
root@PicOS-OVS# ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
device br0 entered promiscuous mode
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/1 vlan_mode=access tag=1 -- set
Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/2 vlan_mode=access tag=1 -- set
Interface te-1/1/2 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_type=0x08847,nw_src=10.10.3.100,nw_dst=10.10.4.100,dl_vlan=1,mpls_
```

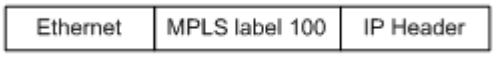
The transmitted packet format to port te-1/1/2 is shown as following (egress):

(3) Configure Switch-C

In switch-C, you need configure one flow which will SWAP the MPLS Label 10 to 100 for traffic RED.

```
root@PicOS-OVS# ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
device br0 entered promiscuous mode
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/1 vlan_mode=access tag=1 -- set
Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/2 vlan_mode=access tag=1 -- set
Interface te-1/1/2 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=1,dl_type=0x08847,nw_src=10.10.1.100,nw_dst=10.10.2.100,dl_vlan=1,mpls_
```

The transmitted packet format to port te-1/1/2 is shown as following (egress):

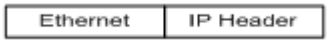


(4) Configure Switch-D

In switch-D, you need configure two flow which will POP the MPLS Label 100 and 200 for traffic RED and BLUE respectively.

```
root@PicOS-OVS# ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8
device br0 entered promiscuous mode
root@PicOS-OVS#
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/1 vlan_mode=access tag=1 -- set
Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/2 vlan_mode=access tag=1 -- set
Interface te-1/1/2 type=pica8
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/3 vlan_mode=access tag=1 -- set
Interface te-1/1/3 type=pica8
root@PicOS-OVS# ovs-vsctl add-port br0 te-1/1/4 vlan_mode=access tag=1 -- set
Interface te-1/1/4 type=pica8
root@PicOS-OVS#
root@PicOS-OVS# ovs-ofctl add-flow br0
in_port=4,dl_type=0x08847,nw_src=10.10.1.100,nw_dst=10.10.2.100,dl_vlan=1,actio
ovs-ofctl add-flow br0
in_port=3,dl_type=0x08847,nw_src=10.10.3.100,nw_dst=10.10.4.100,dl_vlan=1,actio
```

The transmitted packet format to port te-1/1/1 and te-1/1/2 is shown as following (egress):



Multiple Virtual Bridges

In PicOS OVS, you can create multiple virtual bridges that are independent to each other. One physical port is able to add into only one virtual bridge. Each virtual bridge can be configured a controller respectively.

```
root@PicOS-OVS# ovs-vsctl  add-br br0 -- set bridge br0 datapath_type=pica8
other-config=datapath-id=0000d80aa99aaaaa
device br0 entered promiscuous mode
root@PicOS-OVS# ovs-vsctl  add-port br0 te-1/1/1 vlan_mode=access tag=1 --
set Interface te-1/1/1 type=pica8
root@PicOS-OVS# ovs-vsctl  add-port br0 te-1/1/2 vlan_mode=access tag=1 --
set Interface te-1/1/2 type=pica8
root@PicOS-OVS# ovs-vsctl  set-controller br0 tcp:10.10.50.1:6633
root@PicOS-OVS# ovs-vsctl  add-br br1 -- set bridge br1 datapath_type=pica8
other-config=datapath-id=0000d80bb99bbbbb
device br0 entered promiscuous mode
root@PicOS-OVS# ovs-vsctl  add-port br1 te-1/1/3 vlan_mode=access tag=1 --
set Interface te-1/1/3 type=pica8
root@PicOS-OVS# ovs-vsctl  add-port br1 te-1/1/4 vlan_mode=access tag=1 --
set Interface te-1/1/4 type=pica8
root@PicOS-OVS# ovs-vsctl  set-controller br1 tcp:10.10.50.2:6633
```

SSL Connection to Controller

If user want to create SSL connection with controller in PicOS switch, following these steps:

 # Switch

root@PicOS-OVS#**apt-get install openssl**

Reading package lists... Done

Building dependency tree

Reading state information... Done

Suggested packages:

ca-certificates

The following NEW packages will be installed:

openssl

0 upgraded, 1 newly installed, 0 to remove and 17 not upgraded.

Need to get 696 kB of archives.

After this operation, 1070 kB of additional disk space will be used.

WARNING: The following packages cannot be authenticated!

openssl

Authentication warning overridden.

Get:1 http://ftp.debian.org/debian/ stable/main openssl powerpc 1.0.1e-2 [696 kB]

Fetched 696 kB in 5s (131 kB/s)

Selecting previously unselected package openssl.

(Reading database ... 17049 files and directories currently installed.)

Unpacking openssl (from .../openssl_1.0.1e-2_powerpc.deb) ...

Processing triggers for man-db ...

Setting up openssl (1.0.1e-2) ...

root@PicOS-OVS#ovs-pki init

/ovs/bin/ovs-pki: /ovs/var/lib/openvswitch/pki already exists and --force not specified

root@PicOS-OVS#**ovs-pki init --force**

Creating controllerca...

Creating switchca...

root@PicOS-OVS#**cd /ovs/var/lib/openvswitch/pki/controllerca**

root@PicOS-OVS#**ovs-pki req+sign ctl controller**

ctl-req.pem Mon Jan 13 03:26:05 UTC 2014

fingerprint 1cbf63b21301f33d9b4aa30540bff492f15bced3

root@PicOS-OVS#ls

ca.cnf careq.pem crl ctl-cert.pem ctl-req.pem index.txt.attr index.txt.old private serial.old

cacert.pem certs crlnumber ctl-privkey.pem index.txt index.txt.attr.old newcerts serial

root@PicOS-OVS#ls ctl-privkey.pem ctl-cert.pem

ctl-cert.pem ctl-privkey.pem

root@PicOS-OVS#**cd /ovs/var/lib/openvswitch/pki/switchca**

root@PicOS-OVS#**ovs-pki req+sign sc switch**

sc-req.pem Mon Jan 13 03:26:54 UTC 2014

fingerprint 65ed449bee94b8e7b8ba7da6f6584afd2f9cc2fb

root@PicOS-OVS#ls sc-privkey.pem sc-cert.pem

sc-cert.pem sc-privkey.pem

root@PicOS-OVS#

root@PicOS-OVS#**scp /ovs/var/lib/openvswitch/pki/controllerca/ctl-cert.pem 10.10.50.41:/home/build**

The authenticity of host '10.10.50.41 (10.10.50.41)' can't be established.

ECDSA key fingerprint is e6:04:3b:c8:24:36:c7:dd:c1:06:6a:69:e2:3b:82:2f.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added '10.10.50.41' (ECDSA) to the list of known hosts.

root@10.10.50.41's password:

ctl-cert.pem 100% 4063 4.0KB/s 00:00

root@PicOS-OVS#**scp /ovs/var/lib/openvswitch/pki/controllerca/ctl-privkey.pem 10.10.50.41:/home/build**

root@10.10.50.41's password:

ctl-privkey.pem 100% 1675 1.6KB/s 00:00

root@PicOS-OVS#**scp /ovs/var/lib/openvswitch/pki/switchca/cacert.pem 10.10.50.41:/home/build**

root@10.10.50.41's password:

cacert.pem 100% 4028 3.9KB/s 00:00

root@PicOS-OVS#**ovs-vsctl set-ssl /ovs/var/lib/openvswitch/pki/switchca/sc-privkey.pem /ovs/var/lib/openvswitch/pki/switchca/sc-cert.pem /ovs/var/lib/openvswitch/pki/controllerca/cacert.pem**

root@PicOS-OVS#**ovs-vsctl del-br br0**

ovs-vsctl: no bridge named br0

root@PicOS-OVS#**ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8**

root@PicOS-OVS#**ovs-vsctl set-controller br0** ssl:10.10.50.41:6633

root@PicOS-OVS#

 # Controllr

root@dev-41:/home/build# **ryu-manager --ctl-privkey ./ctl-privkey.pem --ctl-cert ./ctl-cert.pem --ca-certs ./cacert.pem --verbose**

loading app ryu.controller.ofp_handler

instantiating app ryu.controller.ofp_handler of OFPHandler

BRICK ofp_event

CONSUMES EventOFPPortDescStatsReply

CONSUMES EventOFPSwitchFeatures

CONSUMES EventOFPErrorMsg

CONSUMES EventOFPEchoRequest

CONSUMES EventOFPHello

connected socket:<eventlet.green.ssl.GreenSSLSocket object at 0x9f1ebfc>
address:('10.10.50.155', 48508)

hello ev <ryu.controller.ofp_event.EventOFPHello object at 0x9ecf1ec>

move onto config mode

switch features ev version: 0x4 msg_type 0x6 xid 0xa2f1cf23
OFPSwitchFeatures(auxiliary_id=0,capabilities=7,datapath_id=74613683395968570 98L,n_buffers=

move onto main mode

Create SSL connection with controller

User want to create SSL connection with controller in PicOS switch

Step-by-step guide

 # Switch

root@PicOS-OVS#**apt-get install openssl**

Reading package lists... Done

Building dependency tree

Reading state information... Done

Suggested packages:

ca-certificates

The following NEW packages will be installed:

openssl

0 upgraded, 1 newly installed, 0 to remove and 17 not upgraded.

Need to get 696 kB of archives.

After this operation, 1070 kB of additional disk space will be used.

WARNING: The following packages cannot be authenticated!

openssl

Authentication warning overridden.

Get:1 http://ftp.debian.org/debian/ stable/main openssl powerpc 1.0.1e-2 [696 kB]

Fetched 696 kB in 5s (131 kB/s)

Selecting previously unselected package openssl.

(Reading database ... 17049 files and directories currently installed.)

Unpacking openssl (from .../openssl_1.0.1e-2_powerpc.deb) ...

Processing triggers for man-db ...

Setting up openssl (1.0.1e-2) ...

root@PicOS-OVS#ovs-pki init

/ovs/bin/ovs-pki: /ovs/var/lib/openvswitch/pki already exists and --force not specified

root@PicOS-OVS#**ovs-pki init --force**

Creating controllerca...

Creating switchca...

root@PicOS-OVS#**cd /ovs/var/lib/openvswitch/pki/controllerca**

root@PicOS-OVS#**ovs-pki req+sign ctl controller**

ctl-req.pem Mon Jan 13 03:26:05 UTC 2014

fingerprint 1cbf63b21301f33d9b4aa30540bff492f15bced3

root@PicOS-OVS#ls

ca.cnf careq.pem crl ctl-cert.pem ctl-req.pem index.txt.attr index.txt.old private serial.old

cacert.pem certs crlnumber ctl-privkey.pem index.txt index.txt.attr.old newcerts serial

root@PicOS-OVS#ls ctl-privkey.pem ctl-cert.pem

ctl-cert.pem ctl-privkey.pem

root@PicOS-OVS#**cd /ovs/var/lib/openvswitch/pki/switchca**

root@PicOS-OVS#**ovs-pki req+sign sc switch**

sc-req.pem Mon Jan 13 03:26:54 UTC 2014

fingerprint 65ed449bee94b8e7b8ba7da6f6584afd2f9cc2fb

root@PicOS-OVS#ls sc-privkey.pem sc-cert.pem

sc-cert.pem sc-privkey.pem

root@PicOS-OVS#

root@PicOS-OVS#**scp /ovs/var/lib/openvswitch/pki/controllerca/ctl-cert.pem 10.10.50.** 41:/home/build

The authenticity of host '10.10.50.41 (10.10.50.41)' can't be established.

ECDSA key fingerprint is e6:04:3b:c8:24:36:c7:dd:c1:06:6a:69:e2:3b:82:2f.

Are you sure you want to continue connecting (yes/no)? yes

Warning: Permanently added '10.10.50.41' (ECDSA) to the list of known hosts.

root@10.10.50.41's password:

ctl-cert.pem 100% 4063 4.0KB/s 00:00

root@PicOS-OVS#**scp /ovs/var/lib/openvswitch/pki/controllerca/ctl-privkey.pem 10.10.50.** 41:/home/build

root@10.10.50.41's password:

ctl-privkey.pem 100% 1675 1.6KB/s 00:00

root@PicOS-OVS#**scp /ovs/var/lib/openvswitch/pki/switchca/cacert.pem 10.10.50.** 41:/home/build

root@10.10.50.41's password:

cacert.pem 100% 4028 3.9KB/s 00:00

root@PicOS-OVS#**ovs-vsctl set-ssl /ovs/var/lib/openvswitch/pki/switchca/sc-privkey.pem /ovs/var/lib/openvswitch/pki/switchca/sc-cert.pem /ovs/var/lib/openvswitch/pki/controllerca/cacert.pem**

root@PicOS-OVS#**ovs-vsctl del-br br0**

ovs-vsctl: no bridge named br0

root@PicOS-OVS#**ovs-vsctl add-br br0 -- set bridge br0 datapath_type=pica8**

root@PicOS-OVS#**ovs-vsctl set-controller br0** ssl:10.10.50.41:6633

root@PicOS-OVS#

 # Controllr

root@dev-41:/home/build# **ryu-manager --ctl-privkey ./ctl-privkey.pem --ctl-cert ./ctl-cert.pem --ca-certs ./cacert.pem --verbose**

loading app ryu.controller.ofp_handler

instantiating app ryu.controller.ofp_handler of OFPHandler

BRICK ofp_event

CONSUMES EventOFPPortDescStatsReply

CONSUMES EventOFPSwitchFeatures

CONSUMES EventOFPErrorMsg

CONSUMES EventOFPEchoRequest

CONSUMES EventOFPHello

connected socket:<eventlet.green.ssl.GreenSSLSocket object at 0x9f1ebfc> address:('10.10.50.155', 48508)

hello ev <ryu.controller.ofp_event.EventOFPHello object at 0x9ecf1ec>

move onto config mode

switch features ev version: 0x4 msg_type 0x6 xid 0xa2f1cf23 OFPSwitchFeatures(auxiliary_id=0,capabilities=7,datapath_id=7461368339596857098L,n_buffers=

move onto main mode

Related articles

- Page: How to configure GRE in OVS mode
- Page: How to configure IGMP snooping
- Page: How to create inband connection with controller in OVS
- Page: How to create SSL connection with controller in OVS