MuL Controller - Northbound API Description

Table of Contents

1 Topology API

- Provide the whole network graph information (Link + Node).

API	Task
GET {version}/topology	List the neighbor information of all registered switches URL Input: - {version} : NBAPI Version. It is currently '1.0' Output : - dpid : own device's dpid - neighbors : specific device's all neighbor information - status : connected device's information. One of 'switch' or 'external' - port : connected own device's port number - neigh_dpid : connected peer device's dpid - neigh_port : connected peer device's port number ex) [{"neighbors": [{"status": "external", "port": 2}, {"status": "external", "port": 1}], "dpid": "0xb8ca3a62"}, {"neighbors": [{"status": "external", "port": 1}, {"status": "switch", "niegh_port": 3, "port": 2, "neigh_dpid": "0x2"}], "dpid": "0x1"}, {"neighbors": [{"status": "switch", "niegh_port": 1, "port": 1, "neigh_dpid": "0x3"}, {"status": "switch", "niegh_port": 2, "port": 3, "neigh_dpid": "0x1"}, {"status": "external", "port": 2}], "dpid": "0x2"}, {"neighbors": [{"status": "external", "port": 3}, {"status": "external", "port": 4}], "dpid": "0x10001b8ca3a62"}, {"neighbors": [{"status": "switch", "niegh_port": 1, "port": 1, "neigh_dpid": "0x2"}, {"status": "external", "port": 2}], "dpid": "0x3"}]
GET {version}/topology/switch/{dpid} / neighbor	Show neighbor information of specific switch URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number Output: - status : connected device's information. one of 'switch' or 'external' - neigh_port: connected device's port number - neigh_dpid : connected device's dpid - port : own port number ex) [{"status": "external", "port": 1}, {"status": "switch", "niegh_port": 3, "port": 2, "neigh_dpid": "0x2"}]

GET {version}/topology/switch	List of switches URL Input: - {version} : NBAPI Version. It is currently '1.0' output : - switches : list of switches - status : registration status between controller and openflow switch - flows : number of registered entries in flow table - meters : number of registered entries in meter table - group : number of registered entries in group table - dpid : Openflow DPID number - peer : connected controller address(IP:PORT) - ports : number of ports ex) {"switches": [{ status: "Published", flows: "5", meters: "0", groups: "0", dpid: "0x1001", peer: "127.0.0.1:34547", ports: "3" }]}
GET {version}/topology/switch/{dpid}	Show detailed info of switch with {dpid} URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number Output: - n_tables : Number of tables supported by datapath - actions : Bitmap of supported "ofp_action_type"s - of_version : supported openflow version - dpid : Openflow DPID number - capabilities : Bitmap of support "ofp_capabilities" - ports : number of ports - n_buffers: Max packets buffered at once ex) {"n_tables": 255, "actions": 0, "of_version": "1.0", "dpid": "0x1001", "capabilites": 199, "ports": 991, "n_buffers": 256} Reference : OenFlow 1.0 version enum ofp_action_type {

	OFPAT_OUTPUT, /* Output to switch port. */
	OFPAT_SET_VLAN_VID, /* Set the 802.1q VLAN id. */
	OFPAT_SET_VLAN_PCP, /* Set the 802.1q priority. */
	OFPAT_STRIP_VLAN, /* Strip the 802.1q header. */
	OFPAT_SET_DL_SRC, /* Ethernet source address. */
	OFPAT_SET_DL_DST, /* Ethernet destination address. */
	OFPAT_SET_NW_SRC, /* IP source address. */
	OFPAT_SET_NW_DST, /* IP destination address. */
	OFPAT_SET_NW_TOS, /* IP ToS (DSCP field, 6 bits). */
	OFPAT_SET_TP_SRC, /* TCP/UDP source port. */
	OFPAT_SET_TP_DST, /* TCP/UDP destination port. */
	OFPAT_ENQUEUE, /* Output to queue. */
	OFPAT_VENDOR = 0xffff
	}
	
	OenFlow 1.3 version
	enum ofp_action_type {
	OFPAT_OUTPUT = 0, /* Output to switch port. */
	OFPAT_COPY_TTL_OUT = 11, /* Copy TTL "outwards" -- from next-to-outermostto outermost */
	OFPAT_COPY_TTL_IN = 12, /* Copy TTL "inwards" -- from outermost tonext-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 a port */
	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 TLV format. */
	OFPAT_PUSH_PBB = 26, /* Push a new PBB service tag (I-TAG) */
	OFPAT_POP_PBB = 27, /* Pop the outer PBB service tag (I-TAG) */
	OFPAT_EXPERIMENTER = 0xffff
	}
	
	ofp_capabilities :
	OpenFlow 1.0 version
	/* Capabilities supported by the datapath. */
	enum ofp_capabilities {
	OFPC_FLOW_STATS = 1 << 0, /* Flow statistics. */
	OFPC_TABLE_STATS = 1 << 1, /* Table statistics. */
	OFPC_PORT_STATS = 1 << 2, /* Port statistics. */

	OFPC_STP = 1 << 3, /* 802.1d spanning tree. */
	OFPC_RESERVED = 1 << 4, /* Reserved, must be zero. */
	OFPC_IP_REASM = 1 << 5, /* Can reassemble IP fragments. */
	OFPC_QUEUE_STATS = 1 << 6, /* Queue statistics. */
	OFPC_ARP_MATCH_IP = 1 << 7 /* Match IP addresses in ARP pkts. */
	};
	
	OenFlow 1.3 version
	/* Capabilities supported by the datapath. */
	enum ofp_capabilities {
	OFPC_FLOW_STATS = 1 << 0, /* Flow statistics. */
	OFPC_TABLE_STATS = 1 << 1, /* Table statistics. */
	OFPC_PORT_STATS = 1 << 2, /* Port statistics. */
	OFPC_GROUP_STATS = 1 << 3, /* Group statistics. */
	OFPC_IP_REASM = 1 << 5, /* Can reassemble IP fragments. */
	OFPC_QUEUE_STATS = 1 << 6, /* Queue statistics. */
	OFPC_PORT_BLOCKED = 1 << 8 /* Switch will block looping ports. */
	}

GET {version}/topology/switch/{dpid}/port	List ports of switch URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number Output: - ports : list of ports - hw_addr : port mac address - state : operational link status (LINK_UP/LINK_DOWN) - curr : current port's "ofp_port_features" - name : interface name - advertised : port's advertised "ofp_port_features" - peer : peer port's received "ofp_port_features" - supported : port's supported "ofp_port_features" - config : administrative link status (PORT_UP/PORT_DOWN) - port_no : port number in the switch ex) {"ports": [{"hw_addr": "FA:DD:1A:E9:98:17", "state": "LINK_UP", "curr": null, "name": "Port3", "advertised": null, "peer": null, "supported": null, "config": "PORT_UP", "port_no": 3}, {"hw_addr": "22:C6:20:AC:51:FE", "state": "LINK_UP", "curr": null, "name": "Port4", "advertised": null, "peer": null, "supported": null, "config": "PORT_UP", "port_no": 4}]} Reference : OenFlow 1.0 version /* Features of physical ports available in a datapath. */

	enum ofp_port_features {
	OFPPF_10MB_HD = 1 << 0, /* 10 Mb half-duplex rate support. */
	OFPPF_10MB_FD = 1 << 1, /* 10 Mb full-duplex rate support. */
	OFPPF_100MB_HD = 1 << 2, /* 100 Mb half-duplex rate support. */
	OFPPF_100MB_FD = 1 << 3, /* 100 Mb full-duplex rate support. */
	OFPPF_1GB_HD = 1 << 4, /* 1 Gb half-duplex rate support. */
	OFPPF_1GB_FD = 1 << 5, /* 1 Gb full-duplex rate support. */
	OFPPF_10GB_FD = 1 << 6, /* 10 Gb full-duplex rate support. */
	OFPPF_COPPER = 1 << 7, /* Copper medium. */
	OFPPF_FIBER = 1 << 8, /* Fiber medium. */
	OFPPF_AUTONEG = 1 << 9, /* Auto-negotiation. */
	OFPPF_PAUSE = 1 << 10, /* Pause. */
	OFPPF_PAUSE_ASYM = 1 << 11 /* Asymmetric pause. */
	};
	
	OenFlow 1.3 version
	/* Features of ports available in a datapath. */
	enum ofp_port_features {
	OFPPF_10MB_HD = 1 << 0, /* 10 Mb half-duplex rate support. */
	OFPPF_10MB_FD = 1 << 1, /* 10 Mb full-duplex rate support. */
	OFPPF_100MB_HD = 1 << 2, /* 100 Mb half-duplex rate support. */
	OFPPF_100MB_FD = 1 << 3, /* 100 Mb full-duplex rate support. */
	OFPPF_1GB_HD = 1 << 4, /* 1 Gb half-duplex rate support. */
	OFPPF_1GB_FD = 1 << 5, /* 1 Gb full-duplex rate support. */
	OFPPF_10GB_FD = 1 << 6, /* 10 Gb full-duplex rate support. */
	OFPPF_40GB_FD = 1 << 7, /* 40 Gb full-duplex rate support. */
	OFPPF_100GB_FD = 1 << 8, /* 100 Gb full-duplex rate support. */
	OFPPF_1TB_FD = 1 << 9, /* 1 Tb full-duplex rate support. */
	OFPPF_OTHER = 1 << 10, /* Other rate, not in the list. */
	OFPPF_COPPER = 1 << 11, /* Copper medium. */
	OFPPF_FIBER = 1 << 12, /* Fiber medium. */
	OFPPF_AUTONEG = 1 << 13, /* Auto-negotiation. */
	OFPPF_PAUSE = 1 << 14, /* Pause. */
	OFPPF_PAUSE_ASYM = 1 << 15 /* Asymmetric pause. */
	};

GET {version}/topology/switch/{dpid} /port/{port_id}	Show detailed info of port URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number - {port_id} : Port number Output: - hw_addr : port mac address - state : operational link status (LINK_UP/LINK_DOWN) - curr : current port's "ofp_port_features" - name : interface name - advertised : port's advertised "ofp_port_features"

	- peer : peer port's received "ofp_port_features" - supported : port's supported "ofp_port_features" - config : administrative link status (PORT_UP/PORT_DOWN) - port_no : port number in the switch ex) {"hw_addr": "FA:DD:1A:E9:98:17", "state": "LINK_UP", "curr": null, "name": "Port3", "advertised": null, "peer": null, "supported": null, "config": "PORT_UP", "port_no": 3}
GET {version}/topology/switch/{dpid}/table/{table_id}	Show table switch features URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number - {table_id} : flow table id Output: - instruction : - instruction_miss : - next_table : - next_table_miss : - write_actions : - write_actions_miss : - apply_actions : - apply_actions_miss : - set_field : - set_field_miss : - apply_set_filed : - apply_set_field_miss : ex) {"instruction ": ["inst-goto", "inst-metadata....], "instruction_miss":["inst-goto", "inst-metadata....] , "next_table":[0, 1, 2...], "next_table_miss":[0,1,2...], "write_actions":["act-output", "act-copy-ttl-out"....], "write_actions_miss":["act-output"...], "apply_actions":["act-output", "act-copy-ttl-out"....], "apply_actions_miss":["act-output"...], "set_field":["in-port","eth-dst",...."mpls-label"], "set_field_miss":["in-port","eth-dst",...."mpls-label"], "apply_set_field":["in-port","eth-dst",...."mpls-label"], "apply_set_field_miss":["in-port",...."mpls-label"]} Reference : Instruction type (OFPIT_XXX) - inst-goto - inst-metadata

	- inst-write-act - inst-apply-act - inst-clear-act - inst-meter action type - act-output - act-copy-ttl-out - act-copy-ttl-in - act-mpls-ttl - act-mpls-dec-ttl - act-push-vlan - act-pop-vlan - act-push-mpls - act-pop-mpls - act-set-queue - act-set-group - act-set-nw-ttl - act-dec-nw-ttl - act-set-field - act-push-pbb - act-pbb Set field type (OFPXMT_OFB_XXX) - in-port - in-phy-port - metadata - eth-dst - eth-src - eth-type - vlan-vid - vlan-pcp - ip-dscp - ip-ecn - ip-proto - ipv4-src - ipv4-dst - tcp-src - tcp-dst - udp-src - udp-dst - sctp-src - sctp-dst - icmp4-type - icmp4-code - arp-opcode - arp-ipv4-src

	- arp-ipv4-dst - arp-src-mac - arp-dst-mac - ipv6-src - ipv6-dst - ipv6-fl-label - icmpv6-type - icmpv6-code - ipv6-nd-target - ipv6-nd-sll - ipv6-nd-tll - mpls-label - mpls-tc - mpls-bos - pbb-isid - tun-id

GET
{version}/topology/switch/{dpid}
/meter
	Show switch meter features
	URL Input:
	- {version} : NBAPI Version. It is currently '1.0'
	- {dpid} : Openflow DPID number
	
	Output:
	- max-bands :
	- bands : supported band types in "ofp_meter_band_type"
	- band-drop
	- max-meter : maximum meter value
	- flags : supported flags types in "ofp_meter_flags"
	- max-color
	
	ex)
	{"max-bands": 255, "bands": ["band-drop", "band-dscp-mark"],
	"max-meter": 16777216, "flags": ["meter-kbps", "meter-pps",
	"meter-burst", "meter_stats"], "max-color": 0}
	
	Reference :
	
	/* Meter band types */
	enum ofp_meter_band_type {
	OFPMBT_DROP = 1, /* Drop packet. */
	OFPMBT_DSCP_REMARK = 2, /* Remark DSCP in the IP header. */
	OFPMBT_EXPERIMENTER = 0xFFFF /* Experimenter meter band. */
	};
	/* Meter configuration flags */
	enum ofp_meter_flags {
	OFPMF_KBPS = 1 << 0, /* Rate value in kb/s (kilo-bit per second). */

	OFPMF_PKTPS = 1 << 1, /* Rate value in packet/sec. */ OFPMF_BURST = 1 << 2, /* Do burst size. */ OFPMF_STATS = 1 << 3, /* Collect statistics. */ }; -
GET {version}/topology/switch/{dpid} /group	Show group features URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number Output: - group_indirect_actions : supported action types when group is indirect mode - group_ff_actions : supported action types when group is fast-failover mode - max_group : maximum entry number - capability : supported group type in ofp_group_capabilies - groups : - group_all_actions: supported action types when group is all mode - group_select_actions : supported action types when group is select mode ex) {"group_indirect_actions": ["act-output", "act-copy-ttl-out",…], "gruop_ff_actions": ["act-output", "act-copy-ttl-out",…], "max_group": [{"all": "16777216"}, {"select": "16777216"}, {"indirect": "16777216"}, {"fast-failover": "16777216"}], "capability": ["grp-flags-select-liveness"], "groups": ["grp-all", "grp-select", "grp-indirect", "grp-fast-failover"], "group_all_actions": ["act-output", "act-copy-ttl-out",…], "group_select_actions": ["act-output", "act-copy-ttl-out",…]} Reference : Group type (OFPGT_XXX) - all - select - indirect - fast-failover capability type (OFPFGC_XXX) - select-weight - select-liveness - chaining - chaining-check

	capability type (OFPFGC_XXX)
	- Same with above
	
	/* Group types. Values in the range [128, 255] are reserved for experimental * use. */
	enum ofp_group_type {
	OFPGT_ALL = 0, /* All (multicast/broadcast) group. */
	OFPGT_SELECT = 1, /* Select group. */
	OFPGT_INDIRECT = 2, /* Indirect group. */
	OFPGT_FF = 3, /* Fast failover group. */
	};
	
	/* Group configuration flags */
	enum ofp_group_capabilities {
	OFPGFC_SELECT_WEIGHT = 1 << 0, /* Support weight for select groups */
	OFPGFC_SELECT_LIVENESS = 1 << 1, /* Support liveness for select groups */
	OFPGFC_CHAINING = 1 << 2, /* Support chaining groups */
	OFPGFC_CHAINING_CHECKS = 1 << 3, /* Check chaining for loops and delete */
	};

POST {version}/topology/switch/{dpid}/limit	Configuration of OpenFlow frame dump function URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number Input Structure : - rx : receive frame, integer(0:disable, over 1: enable and configure the packet-in rate-limit) - tx : transmit frame, integer(0:disable, over 1:enable and configure the packet-out rate-limit) ex) {"rx": 10,"tx": 0} Output : - message :string (SUCCESS/FAIL) - rx : Enable / Disable - tx : Enable / Disable ex) <Response [200]> {"rx": "Enable", "tx": "Enable"}
GET	Show the configuration of OpenFlow frame dump function

{version}/topology/switch/{dpid} /limit	URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number Output : - rx : receive frame, integer(0:disable, over 1: enable and configure the packet-in rate-limit) - tx : transmit frame, integer(0:disable, over 1:enable and configure the packet-out rate-limit) ex) {"rx": 10,"tx": 0}

2 Flow Table API

- Provide the forwarding rule management according to the OpenflowVer 1.3 specs.

API	Task
GET {version}/flowtable/{dpid}/flow	List all flows in switch URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number Output: - dpid : Openflow DPID number - flows: List of flows in switch - dl_dst : Destination MAC address - dl_src: source MAC address - dl_type : ether type - dl_vlan : VLAN ID - dl_vlan_pcp : VLAN Priority - mpls_bos : MPLS bos - mpls_tc : MPLS tc - mpls_label : MPLS label - nw_src : source IP address - nw_dst : destination IP address - nw_tos : IP TOS, exactly DSCP 6bit - nw_proto : IP protocol or lower 8 bits of ARP code - tp_src : TCP/UDP source port number - tp_dst : TCP/UDP destination port number - in_port : in coming port number - priority : priority in flow table - instructions : List of instructions in flow - instruction : instruction type - value : value for the instruction(when instruction is METER or GOTO_TABLE) - actions : List of actions in flow(when instruction is WRITE_ACTION or APPLY_ACTION) - action : action name - value : value for the action - stat : statistic information of this flow - byte_count : total received byte count - packet_count : total received packet count - alive : time after the flow was created (second) - pps : real time rate (packet per seconds) - bps : real time rate (bits per seconds) - flag : flow's status parameter

	- flow_id : Flow unique id
	ex)
	{
	dl_type: "0x800",
	stat: {
	pps: "0.000000",
	byte_count: 204,
	bps: "0.000000",
	pkt_count: 2,
	alive: 305775
	},
	nw_dst: "10.0.0.8/32",
	flow_id:
	"00000000000000010000000000000000000000000800070e0000000000000a000008200a0000042000000000000000000000100000003"
	,
	priority: 3,
	table_id: 0,
	dpid: "0x1",
	flags: "dynamic no-clone verified non-local clean",
	dl_vlan: 1806,
	nw_src: "10.0.0.4/32",
	in_port: 1,
	instructions: [
	{
	instruction: "APPLY_ACTIONS",
	actions: [
	{
	action: "OUTPUT",
	value: 2
	}
	]
	}
	]
	}

POST {version}/flowtable/{dpid}/flow	Add new flow to flowtable in switch URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number Input structure: - dl_dst : Destination MAC address - dl_src: source MAC address - dl_type : ether type - dl_vlan : VLAN ID - dl_vlan_pcp : VLAN Priority

	- mpls_bos : MPLS bos - mpls_tc : MPLS tc - mpls_label : MPLS label - nw_src : source IP address. ipv4 or ipv6 format. - nw_dst : destination IP address. ipv4 of ipv6 format. - nw_tos : IP TOS, exactly DSCP 6bit - nw_proto : IP protocol or lower 8 bits of ARP code - tp_src : TCP/UDP source port number - tp_dst : TCP/UDP destination port number - in_port : in coming port number - priority : priority in flow table - instructions : List of instructions for this flow - instruction : instruction type. One of WRITE_ACTIONS, APPLY_ACTIONS, METER, GOTO_TABLE - value : value for the instruction(meter_id when instruction is METER, table_id when GOTO_TABLE) - actions : List of actions in flow(when instruction is WRITE_ACTION or APPLY_ACTION) - action : action name, please refer below ACTION_NAME_LIST - value : value for the action - barrier : One of ‘enable’ or ‘disable’. when enabled, send an accompanying barrier after flow-add command - stat : One of ‘enable’ or ‘disable’. when enabled, flow stat info is able to gather ex) {"dl_dst": "x", "ds_src": "x", "nw_dst": "x", "nw_src": "x", "dl_vlan”: "x", "tp_src": "x", "tp_dst": "x","priority": "x", "in_port": "x", “instructions” : [{“type” : “WRITE_ACTIONS”,"actions": [{"action": "OUTPUT", "value": "2"}] }]} Output: - flow_id : created flow id ex) {"flow_id": "050b1dba-984d-4001-8cf4-32bb1e1afc56"} Reference : ACTION_NAME_LIST { 'OUTPUT' : int, 'SET_VLAN_VID' : int 'SET_VLAN_PCP' : int 'STRIP_VLAN' : no value 'SET_DL_SRC' : str, ex)01:02:03:04:05:06 'SET_DL_DST' : str, ex)01:02:03:04:05:06

	'SET_NW_SRC' : str, ex)1.1.1.1 'SET_NW_DST' : str, ex)1.1.1.1 'SET_NW_SRC6' : str, ex)1:1:1:1:1:1:1:1 'SET_NW_DST6' : str, ex)1:1:1:1:1:1:1:1 'SET_NW_TOS' : int <0-63> 'SET_TP_SRC' : int 'SET_TP_DST' : int 'CP_TTL_OUT' : no value 'CP_TTL_IN' : no value 'SET_MPLS_TTL': int, default value = 0 'DEC_MPLS_TTL': no value 'PUSH_VLAN' : no value 'PUSH_SVLAN' : no value 'POP_VLAN' : no value 'STRIP_VLAN' : no value 'PUSH_MPLS' : int, default value = 0 'POP_MPLS' : int, default value = 0 'SET_QUEUE' : int, default value = 0 'GROUP' : int, default value = 0 'SET_NW_TTL' : int, default value = 0 'DEC_NW_TTL' : no value 'SET_ETH_TYPE' : str, ex)0x0800 'SET_MPLS_LABEL' : int, default value = 0 'SET_MPLS_TC' : int, default value = 0 'SET_MPLS_BOS' : int, default value = 0 'PUSH_PBB' : int, default value = 0 'POP_PBB' : no value }
POST {version}/flowtable/{dpid}/flow/{flow_id}	delete specific flow matched {flow_id} and add new flow. just for gui URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number - {flow_id} : Target flow id for delete Input structure: - Please refer above description of " {version}/flowtable/{dpid}/flow " Output: - flow_id : created flow id ex) {"flow_id": "050b1dba-984d-4001-8cf4-32bb1e1afc56"}
GET {version}/flowtable/{dpid}/flow/{flow_id}	Show detailed information of specific flow URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number - {flow_id} : Target flow id

	Output: - Please refer above description of " {version}/flowtable/{dpid}/flow " ex) {priority: 0,byte_count: 0,dpid: "0x1b8ca3a62f744",flow: {dl_type: 2048,ip.nw_dst: "6.6.6.6",mpls_bos: 0,mpls_tc: 0,dl_vlan_pcp: 0,dl_src: "00:AC:AC:AC:AC:AA",nw_proto: 0,table_id: 0,mpls_label: 0,tp_dst: 0,tp_src: 0,ip.nw_src: "6.6.6.6",nw_tos: 0,dl_dst: "00:AC:AC:AC:AC:AA",dl_vlan: 3,in_port: 0},alive: 494,pps: "0.000000",bps: "0.000000",flags: "static no-clone verified non-local",flow_id: "45e82ab8-28e2-4dce-8a7c-0dc323e4b800",pkt_count: 0,instructions: [{type: "WRITE_ACTIONS",actions: [{action: "SET_DL_SRC",value: "0x00:bb:bb:bb:bb:bb"},{action: "OUTPUT",value: 1}]}]}
DELETE {version}/flowtable/{dpid}/flow/{flow_id}	Delete flow from switch URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number - {flow_id} : Flow id Output: - flow_id : Flow id ex) {"flow_id": "00000000000000004fffffffffff0000000000000000000000000000000000000000000000000000000000000000000000000000000000000002"}

3 Group Table API

API	Task
GET {version}/grouptable/{dpid}/group	List all groups in switch URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number Output: - groups: List of groups in switch - group_id : Group identifier

	- type : Determine group semantics - flags : Supported flags types in “ofp_group_type” - byte_count : Number of bytes processed by group - packet_count : Number of packets processed by group - duration_sec : Time group has been alive in seconds - duration_nsec : Time group has been nanoseconds beyond - action-bucket : an ordered list of actions buckets where each action bucket contains a set of actions excuted and associated parameters - actions : List of actions in group - action : action name - value : value for the action ex) {groups: [{packet_count: 0, duration_sec: 0, flags:"Not-verified", byte_count: 0, action-buckets: [{action_bucket: "0",actions: [{action: "SET_DL_DST", value: "0x00:02:04:01:02:01"}, {action: "OUTPUT", value: 1}]}, {action_bucket: "1", actions: [{action: "SET_QUEUE", value:1}]}], group_id: 1, type: "ff", duration_nsec: 0}}
POST {version}/grouptable/{dpid}/group	Add new group to grouptable in switch URL Input: - {version} : NBAPI Version. It is currently ‘1.0’ - {dpid} : Openflow DPID number Input structure: - group_id : Group identifier - type : one of <all|ff|indirect|select> - action_buckets : List of buckets in group - weight : Relative weight of bucket.(only defined for select groups) - ff_port : Port whose state affects whether this bucket is live.(Only required for fast failover groups) - ff_group : Group whose state affects whether this bucket is live. (Only required for fast failover groups) - actions : List of actions in groups

	- action : action name - value : value for the action ex) { "group_id" : 1, "type" : "all", "action_buckets" : [{ "actions" : [{ 'action' : 'SET_DL_DST' , 'value' : '00:01:02:03:04:05' }, { 'action' : 'OUTPUT' , 'value' : 1 }]}]} ex 2) {"group_id" : 2, "type" : "ff", "action_buckets" : [{ 'ff_port' : 1, 'ff_group' : 5, 'actions' : [{ 'action' : 'SET_DL_DST' , 'value' : '00:02:03:01:02:05' },{ 'action' : 'OUTPUT', 'value' : 1 }] }, { 'ff_port' : 2, 'actions' : [{ 'action' : 'OUTPUT', 'value' : 1 }]}]} Output: - group_id : Group id ex) {“group_id” : 1}
GET {version}/grouptable/{dpid}/group/{group_id}	Show detailed information of specific group URL Input: - {version} : NBAPI Version. It is currently ‘1.0’ - {dpid} : Openflow DPID number - {group_id} : Target group id Output: - Please refer above description of “ {version}/grouptable/{dpid}/group “ ex) {packet_count: 0, duration_sec: 0, flags:"Not-verified", byte_count: 0, action-buckets: [{action_bucket: "0",actions: [{action: "SET_DL_DST", value: "0x00:02:04:01:02:01"}, {action: "OUTPUT", value: 1}]}, {action_bucket: "1", actions: [{action: "SET_QUEUE", value:1}]}], group_id: 1, type: "ff", duration_nsec: 0}
DELETE {version}/grouptable/{dpid}/group/{group_id}	Delete group from switch URL Input: - {version} : NBAPI Version. It is currently ‘1.0’ - {dpid} : Openflow DPID number - {group_id} : Group id Output: - group_id : Group id ex) {“group_id”:1}

4 Meter Table API

API	Task
GET {version}/metertable/{dpid}/meter	List all meters in switch URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number Output: - meters: List of flows in switch - meter_id : Meter identifier - type : One of <kbps|pktps> - burst : One of <yes|no> - stats : One of <yes|no> - flow_count : Number of flows bound to meter - byte_in_count : Number of bytes in inpu - packet_in_count : Number of packets in input - duration_sec : Time meter has been alive in seconds - duration_nsec : Time meter has been alive in nanoseconds beyond - meter_bands : - band_type : One of <dscp+remark|drop> - rate : Rate for packets - burst_size : Size of burst - prec_level : Number of precendence level to substract ex)
POST {version}/metertable/{dpid}/meter	Add new group to grouptable in switch URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number Input structure: - meter_id : Meter identifier - type : Type of meter. One of <kbps|pktps> - burst : One of <yes|no> - stats : One of <yes|no> - meter_bands : List of meter band - band_type : One of <dscp_remark|drop> - rate : Rate for packets - burst_size : Size of burst - prec_level : Number of precendence level to substract ex) {meter_id:1, type:kbps, burst:yes, stats:yes, meter_bands:[{band_type:dscp_remark, rate:1024, burst_size:100, prec_level:1},{band_type:drop, rate:2048, burst_size:300}]} Output: - meter_id : Meter id ex) {"meter_id":1}

GET {version}/metertable/{dpid}/meter	Show detailed information of specific group URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number - {meter_id} : Target meter id Output: - Please refer above description of " {version}/metertable/{dpid}/meter " ex)
DELETE {version}/metertable/{dpid}/meter/{meter_id}	Delete meter from switch URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number - {meter_id} : Meter id Output: - meter_id : Meter id ex) {"meter_id":1}

5 Route API

- Implementation routing algorithm over the network graph as seen by the topology manager. Provide the end-to-end path information according to the algorithm.

API	Task
GET {version}/route	List supported routing algorithms in the openflow domain URL Input: - {version} : NBAPI Version. It is currently '1.0': Output: - algorithms : List of the supported routing algorithms ex) {"algorithms": ["warshall", "dijkstra"...]}
GET {version}/route/path	List installed flow path (path between two devices, hop-by-hop) in the openflow domain URL Input: - {version} : NBAPI Version. It is currently '1.0': Output: - src_host : Source device's information - dst_host : Destination device's information - network_id : Network id

	- tenant_id : Tenant_id - switch_dpid : DPID of installed switch - dl_src : Mac address of this host - nw_src : IP address of this host - port : Switch port number connected to this host - route_link : List of the installed paths - to_sw_port : Switch port number connected to next hop - to_switch : DPID of this switch - hop : Number of hops from src_host to this switch ex) routes: [{route_link: [{to_sw_port: "3",hop: "1",to_switch: "0x3"},{to_sw_port: "3",hop: "2",to_switch: "0x2"},{to_sw_port:"2",hop: "3",to_switch: "0x1"},{to_sw_port:"1",hop: "4",to_switch: "0x5"},{to_sw_port:"1",hop: "5",to_switch: "0x6"}],dst_host: {network_id: "00000000-0000-0000-0000-000000000000",tenant_id:"00000000-0000-0000-0000-000000000000",switch_dpid: "0x6",dl_src:"F6:9B:8E:1C:B6:8C",nw_src: "10.0.0.5",port: " 1"},src_host: {network_id:"00000000-0000-0000-0000-000000000000",tenant_id: "00000000-0000-0000-0000-000000000000",switch_dpid: "0x3",dl_src: "AA:E6:54:E0:1E:74",nw_src: "10.0.0.1",port: " 1"}}}
POST {version}/route/path	Get list of flow path destination to specific host(path between two devices, hop-by-hop) in the openflow domain URL Input: - {version} : NBAPI Version. It is currently '1.0': Input structure: - tenant_id : host's tenant id. uuid format. - network_id : host's network id. uuid format. - dl_dst : host's mac address - nw_dst : host's ip address, with no mask. Output: - src_host : Source device's information - dst_host : Destination device's information - network_id : Network id - tenant_id : Tenant_id - switch_dpid : DPID of installed switch - dl_src : Mac address of this host - nw_src : IP address of this host - port : Switch port number connected to this host - route_link : List of the installed paths - to_sw_port : Switch port number connected to next hop - to_switch : DPID of this switch - hop : Number of hops from src_host to this switch ex) {"routes": [{"route_link": [{"to_sw_port": "3", "hop": "1",

	"to_switch": "0x3"}, {"to_sw_port": "2", "hop": "2", "to_switch": "0x2"}, {"to_sw_port": "1", "hop": "3", "to_switch": "0x4"}], "dst_host": {"network_id": "00000000-0000-0000-0000-000000000000", "tenant_id": "00000000-0000-0000-0000-000000000000", "switch_dpid": "0x4", "dl_src": "F6:9F:C6:94:AA:8C", "nw_src": "10.0.0.3", "port": "1"}, "src_host": {"network_id": "00000000-0000-0000-0000-000000000000", "tenant_id": "00000000-0000-0000-0000-000000000000", "switch_dpid": "0x3", "dl_src": "46:B6:92:1B:B7:69", "nw_src": "10.0.0.1", "port": "1"}}]}
POST {version}/route/path	Get list of flow path source from specific host(path between two devices, hop-by-hop) in the openflow domain URL Input: - {version} : NBAPI Version. It is currently '1.0': Input structure: - tenant_id : host's tenant id. uuid format. - network_id : host's network id. uuid format. - dl_src: host's mac address - nw_src : host's ip address, with no mask. Output: - src_host : Source device's information - dst_host : Destination device's information - network_id : Network id - tenant_id : Tenant_id - switch_dpid : DPID of installed switch - dl_src : Mac address of this host - nw_src : IP address of this host - port : Switch port number connected to this host - route_link : List of the installed paths - to_sw_port : Switch port number connected to next hop - to_switch : DPID of this switch - hop : Number of hops from src_host to this switch ex) {"routes": [{"route_link": [{"to_sw_port": "3", "hop": "1", "to_switch": "0x3"}, {"to_sw_port": "2", "hop": "2", "to_switch": "0x2"}, {"to_sw_port": "1", "hop": "3", "to_switch": "0x4"}], "dst_host": {"network_id": "00000000-0000-0000-0000-000000000000", "tenant_id": "00000000-0000-0000-0000-000000000000", "switch_dpid": "0x4", "dl_src": "F6:9F:C6:94:AA:8C", "nw_src": "10.0.0.3", "port": "1"}, "src_host": {"network_id": "00000000-0000-0000-0000-000000000000", "tenant_id": "00000000-0000-0000-0000-000000000000", "switch_dpid": "0x3", "dl_src": "46:B6:92:1B:B7:69", "nw_src": "10.0.0.1", "port": "1"}}]}
POST {version}/route/path	Get a flow path between two specific host(path between two devices, hop-by-hop) in the openflow domain URL Input: - {version} : NBAPI Version. It is currently '1.0':

	Input structure: - tenant_id : host's tenant id. uuid format. - network_id : host's network id. uuid format. - dl_dst : dst host's mac address - nw_src : dst host's ip address, with no mask. - dl_src : src host's mac address - nw_dst : src host's ip address, with no mask. Output: - src_host : Source device's information - dst_host : Destination device's information - network_id : Network id - tenant_id : Tenant_id - switch_dpid : DPID of installed switch - dl_src : Mac address of this host - nw_src : IP address of this host - port : Switch port number connected to this host - route_link : List of the installed paths - to_sw_port : Switch port number connected to next hop - to_switch : DPID of this switch - hop : Number of hops from src_host to this switch ex) {"routes": [{"route_link": [{"to_sw_port": "3", "hop": "1", "to_switch": "0x3"}, {"to_sw_port": "2", "hop": "2", "to_switch": "0x2"}, {"to_sw_port": "1", "hop": "3", "to_switch": "0x4"}], "dst_host": {"network_id": "00000000-0000-0000-0000-000000000000", "tenant_id": "00000000-0000-0000-0000-000000000000", "switch_dpid": "0x4", "dl_src": "F6:9F:C6:94:AA:8C", "nw_src": "10.0.0.3", "port": "1"}, "src_host": {"network_id": "00000000-0000-0000-0000-000000000000", "tenant_id": "00000000-0000-0000-0000-000000000000", "switch_dpid": "0x3", "dl_src": "46:B6:92:1B:B7:69", "nw_src": "10.0.0.1", "port": "1"}}]}
GET {version}/route/path/{src_dpid}/{dst_dpid}	Show detailed info of simple path URL Input: - {version} : NBAPI Version. It is currently '1.0' - {src_dpid} : DPID of source switch - {dst_dpid} : DPID of destination switch. start with '0x' Output: - to_sw_port : Switch port number connected to next hop - to_switch : DPID of this switch - hop : Number of hops from src_host to this switch ex) {path: [{port: 3,hop: 0,to_switch: "0x2"},{port: 2,hop: 1,to_switch: "0x1"},{port: null,hop: 2,to_switch: "0x5"}]}
GET {version}/route/path/{src_dpid}/{src_port}/{dst_dpid}/{d	Show End To End paths URL Input:

st_port}	- {version} : NBAPI Version. It is currently '1.0' - {src _dpid} : DPID of first hop switch - {src_port} : Ingress port of first hop switch - {dst_dpid} : DPID of last hop switch - {dst_port} : Ougress port of last hop switch Output: - hops : List of the hops of the path(hop_count:0 is first and hop_count:1 is next hop, and so on) - dpid : DPID in this hop - ingress_port: ingress port number in this hop - outgress_port: egress port number in this hop ex) {"hops": [{"dpid": 1, ingress_port": 1, "outgress_port": 3}, {"dpid": 1, ingress_port": 1, "outgress_port": 3 }]}
PUT {version}/route/path/{path_id}	Modify flow path URL Input: - {version} : NBAPI Version. It is currently '1.0': - {path_id} : Path id Input: - src_dev_id : Source Device Id - dst_dev_id : Destination Device Id - algorithm : PCE algorithm ex) {"src_dev_id" : 1, "dst_dev_id" : 2, "algorithm":" warshall "} Ourput: - path_id : Path id ex) {"path _id" : 1}
DELETE /route/path/{id}	Remove simple path from system URL Input: - {version} : NBAPI Version. It is currently '1.0': - {path_id} : Path id Ourput: - path_id : Path id ex) {"path _id" : 1}
GET /path/servicechain	List installed service chain (path with sequence of devices to visit)
POST /path/ servicechain	Add and install new service chain
GET /path/ servicechain /{id}	Show detailed info of service chain

PUT /path/ servicechain /{id}	Modify service chain
DELETE /path/ servicechain /{id}	Remove service chain from system

6 Fabric (Virtual Network) API

- Provide the multi-tenancy. Tenant ID defines the tenant domain and Network ID defines the network domains(for example, IP/SUBNET domain).

- Provide the host joining function in the specific tenancy domain(tenant ID + network ID). Joined hosts are automatically connected by the Floyd-Warshall algorithm.

API	Task
GET {version}/fabric/network	List virtual networks
POST {version}/fabric/network	Add and install new virtual network
GET {version}/fabric/network/{id}	Show detailed info of virtual network
PUT {version}/fabric/network/{id}	Modify virtual network
DELETE {version}/fabric/network/{id}	Remove virtual network from system
GET {version}/fabric/subnet	List subnets
POST {version}/fabric/subnet	Add and install new subnet
GET {version}/fabric/subnet /{id}	Show detailed info of subnet
PUT {version}/fabric/subnet /{id}	Modify subnet
DELETE /fabric/subnet /{id}	Remove subnet from system
GET {version}/fabric/host	List of all Fabric Host Devices URL Input: - {version} : NBAPI Version. It is currently '1.0' Output: - hosts : List of the registered fabric hosts - nw_src : IP address of the registered fabric hosts - dl_src : Mac address of the registered fabric hosts - dpid : openflowdpid which is connected by the host - in_port : the port number of openflowswich which is connected by the host ex) {hosts: [{network_id:"00000000-0000-0000-0000-000000000002",tenant_id: "01000000-0000-0000-0000-000000000001",switch_dpid: "0x7",dl_src: "0E:4B:D3:84:99:55",nw_src: "10.0.0.7",port: " 1"}}
GET {version}/fabric/tenant/{tenant_id}/network/{network_id}/hos	List of Fabric Host Devices URL Input: - {version} : NBAPI Version. It is currently '1.0'

t	- {tenant_id} : Tenant_id - {network_id} : Network_id * Tenant_id and Network support the multi-tenancy. Network represents as like the subnet domain. Output: - hosts : List of the registered fabric hosts - nw_src : IP address of the registered fabric hosts - dl_src : Mac address of the registered fabric hosts - dpid : openflowdpid which is connected by the host - in_port : the port number of openflowswich which is connected by the host ex) {"hosts": ["dl_src": 1, “nw_src": 1, , "dpid": 1", “in_port": 1}]}
GET {version}/fabric/tenant/{tenant_id}/network /{network_id}/host/{host_id}	Show info of specific Fabric Host Device URL Input: - {version} : NBAPI Version. It is currently ‘1.0’ - {tenant_id} : Tenant id - {network_id} : Network id - {host_id} : IP address of specific fabric host * Tenant_id and Network support the multi-tenancy. Network represents as like the subnet domain. Output: - hosts : List of the registered fabric hosts - nw_src : IP address of the registered fabric hosts - dl_src : Mac address of the registered fabric hosts - dpid : openflowdpid which is connected by the host in_port : the port number of openflowswich which is connected by the host ex) {"dl_src": 1, “nw_src": 1, , "dpid": 1", “in_port": 1}
POST {version}/fabric/ tenant /{tenant_id}/network /{network_id}/host	Add Fabric host for the non-gateway mode Show detailed info of Fabric Host Device URL Input: - {version} : NBAPI Version. It is currently ‘1.0’ - {tenant_id} : Tenant id - {network_id} : Network id * Tenant_id and Network support the multi-tenancy. Network represents as like the subnet domain. Input structure: - nw_src : IP address of the registered fabric hosts - dl_Src: Mac address of the registered fabric hosts - dpid : openflowdpid which is connected by the host - in_port : the port number of openflowswich which is connected by the host

	- is_gw : GW Mode or Non GW Mode ex) {"nw_src": 1, "dl_src": 1, "host_ip": 1, , "dpid": 1", “in_port": 1, “is_gw”:yes} Output: - host_id : IP address of the registered fabric hosts ex) {“host_id”:”x”}
PUT {version}/fabric/ tenant /{tenant_id}/network /{network_id}/host/{host_id}	Modify Fabric Host Device URL Input: - {version} : NBAPI Version. It is currently ‘1.0’ - {tenant_id} : Tenant id - {network_id} : Network id - {host_id} : IP address of the registered fabric hosts Input structure: - host_id : IP address of the registered fabric hosts - host_mac : Mac address of the registered fabric hosts - dpid : openflowdpid which is connected by the host - port : the port number of openflowswich which is connected by the host ex) {"host_id": 1, "host_mac": 1, "host_ip": 1, , "dpid": 1", port": 1} Output: - host_id : IP address of the registered fabric hosts ex) {“host_id”:”x”}
DELETE {version}/fabric/ tenant /{tenant_id}/network /{network_id}/host/{host_id}	Delete Fabric host URL Input: - {version} : NBAPI Version. It is currently ‘1.0’ - {tenant_id} : Tenant id - {network_id} : Network id - {host_id} : IP address of the registered fabric hosts Output: - host_id : IP address of the registered fabric hosts ex) {“host_id”:”x”}

7 Openstack API

Just for support openstack (under development)

API	Task
GET /1.0/openstack/networks	List of Openstack networks URL Input: - {version} : NBAPI Version. It is currently '1.0' Output: - tenant_and_networks : List of the registered openstack tenant and network - tenant_id : Id of tenant. uuid formed. - network_id : Id of network. uuid formed ex) {tenant_and_networks: [{network_id:"00000000-0000-0000-0000-000000000000",tenant_id:"00000000-0000-0000-0000-000000000000"}]}
GET /1.0/openstack/host	List of all Openstack Host Devices URL Input: - {version} : NBAPI Version. It is currently '1.0' * Tenant_id and Network support the multi-tenancy. Network represents as like the subnet domain. Output: - hosts : List of the registered fabric hosts - nw_src : IP address of the registered fabric hosts - dl_src : Mac address of the registered fabric hosts - dpid : openflowdpid which is connected by the host - in_port : the port number of openflowswich which is connected by the host ex) {"hosts": ["dl_src": 1, "nw_src": 1, , "dpid": 1", "in_port": 1}]}
POST /1.0/openstack/tenant/{tenant_id}/network/{network_id}/host	Add Fabric host for the non-gateway mode Show detailed info of Fabric Host Device URL Input: - {version} : NBAPI Version. It is currently '1.0' - {tenant_id} : Tenant id - {network_id} : Network id * Tenant_id and Network support the multi-tenancy. Network represents as like the subnet domain. Input structure:

	- nw_src : IP address of the registered fabric hosts - dl_Src: Mac address of the registered fabric hosts - dpid : openflowdpid which is connected by the host - in_port : the port number of openflowswich which is connected by the host - is_gw : GW Mode or Non GW Mode ex) {"nw_src": 1, "dl_src": 1, "host_ip": 1, , "dpid": 1", “in_port": 1, “is_gw”:yes} Output: - host_id : IP address of the registered fabric hosts ex) {“host_id”:”x”}
DELETE /1.0/openstack/tenant/{tenant_id}/network/{network_id}/host/{host_id}	Delete Fabric host URL Input: - {version} : NBAPI Version. It is currently ‘1.0’ - {tenant_id} : Tenant id - {network_id} : Network id - {host_id} : IP address of the registered fabric hosts Output: - host_id : IP address of the registered fabric hosts ex) {“host_id”:”x”}

8 Stat API

Provide relevant statistics information.

API	Task
GET {version}/flowtable/{dpid}/flow or {version}/flowtable/{dpid}/flow/{flow_id}	Flow statistics information is included in the flow information. So you can use left flow NB-API. When you create a flow with NBAPI, API enables flow stat function automatically. We will support enable/disable feature with other APIs that can enable/ disable flow stat function per flow. Show all statistics information of a switch or Show the statistics information of specific flow URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number - {flow_id} : target flow id Output: Refer flow API's description
GET {version}/ stats /switch/{dpid}/port/{port_no}	Show every statistics info of switch port with {dpid}, {port_no}. Currently this API enables port stat function and gets stat info also. We will support enable/disable feature with other APIs that can enable/ disable port stat function per port. URL Input: - {version} : NBAPI Version. It is currently '1.0' - {dpid} : Openflow DPID number - {port_no} : Openflow port numbr Output: - tx_dropped : dropped packets in transmitting - rx_packets : total received packets - rx_crc_err : total crc error packets in receiving - tx_bytes : total bytes in transmitting - rx_dropped : dropped packets in receiving - rx_over_err : total over-sized error packets in receiving - rx_frame_err : total frame error packets in receiving

	- rx_bytes : total received bytes - tx_errors : total error packets in transmitting - duration_nsec : nsec time after statistics data had been collected - collisions : total collision error packets - duration_sec : sec time after statistics data had been collected - rx_errors : total error packets in receiving - tx_packets : total transmitted packets ex) {"tx_dropped": 0, "rx_packets": 0, "rx_crc_err": 0, "tx_bytes": 25183080, "rx_dropped": 0, "port_no": 3, "rx_over_err": 0, "rx_frame_err": 0, "rx_bytes": 0, "tx_errors": 0, "duration_nsec": 916241000, "collisions": 0, "duration_sec": 779692, "rx_errors": 0, "tx_packets": 699530}

9 Dashboard API

API	Task

GET {version}/dashboard	Show mul-server(server that mul running on)'s status. Such as cpu usage, memory usage and so on. URL Input: - {version} : NBAPI Version. It is currently '1.0' Output: -cpus : list of cpus -cpu_num : cpu number -cpu_percent : percentage of specific cpu usage -mem_percent : percentage of memory usage of this server -load_average : average of system load. 1.0 on a single core cpu represents 100% utilization. If server has 8 cpus, 8.0 represent 100% utilization. Returns 3 value, each stands last 1, 5, 15 minutes load average. -uptime : time that this server has booted on -mul_process : process of mul or mul-applications -pname : name of this process -virt : virtual size of this process(byte) -res : resident size of this process(byte) -cpu_percent : percentage of this process's cpu usage -mem_percent : percentage of this process's memory usage ex){"load_average": [0.01, 0.03, 0.05], "mem_percent": 9.0, "mul_process": [{"mem_percent": 0.008570177053509646, "p_status": "running", "cpu_percent": 0.0, "virt": 103559168, "pname": "mulcli", "res": 1437696}, {"mem_percent": 0.008106264335513398, "p_status": "running", "cpu_percent": 0.0, "virt": 27934720, "pname": "mull2sw", "res": 1359872}, {"mem_percent": 0.15299353110339442, "p_status": "running", "cpu_percent": 0.0, "virt": 52203520, "pname": "multr", "res": 25665536}, {"mem_percent": 0.009400336654134513, "p_status": "running", "cpu_percent": 0.0, "virt": 52232192, "pname": "mulfab", "res": 1576960}, {"mem_percent": 0.009229421442241157, "p_status": "running", "cpu_percent": 0.0, "virt": 52219904, "pname": "mulmakdi", "res": 1548288}, {"mem_percent": 0.02099815460404073, "p_status": "running", "cpu_percent": 0.0, "virt": 799862784, "pname": "mul", "res": 3522560}], "uptime": "4 days, 23:45:37.874561", "cpus": [{"cpu_num": 0, "cpu_percent": 0.0}, {"cpu_num": 1, "cpu_percent": 0.0}, {"cpu_num": 2, "cpu_percent":

	0.0}, {"cpu_num": 3, "cpu_percent": 0.0}, {"cpu_num": 4, "cpu_percent": 0.0}, {"cpu_num": 5, "cpu_percent": 0.0}, {"cpu_num": 6, "cpu_percent": 0.0}, {"cpu_num": 7, "cpu_percent": 0.0}, {"cpu_num": 8, "cpu_percent": 0.0}, {"cpu_num": 9, "cpu_percent": 0.0}, {"cpu_num": 10, "cpu_percent": 0.0}, {"cpu_num": 11, "cpu_percent": 0.0}, {"cpu_num": 12, "cpu_percent": 0.0}, {"cpu_num": 13, "cpu_percent": 0.0}, {"cpu_num": 14, "cpu_percent": 0.0}, {"cpu_num": 15, "cpu_percent": 0.0}]}
POST {version}/dashboard	Start specific mul applications. URL Input: - {version} : NBAPI Version. It is currently '1.0' Input Structure: -name : name of mul application. One of {'mulcli'|'mulfab'|'mull2sw'|'multr'|'mulmakdi'} ex) {'name' : 'mulcli'} Output : - application : string(application name) - status : start/fail ex) {'mulcli' : 'start'}

DELETE {version}/dashboard	Kill specific mul applications. URL Input: - {version} : NBAPI Version. It is currently '1.0' Input Structure: -name : name of mul application ex) {'name' : 'mulcli'} Output : - application : string(application name) - status : killed/fail ex) {'mulcli':'killed'}
GET {version/dashboard/appname	Start specific mul applications. URL Input: - {version} : NBAPI Version. It is currently '1.0' Output: -mul_app_name : name of mul application support this controller. One of {'mulcli'|'mulfab'|'mull2sw'|'multr'|'mulmakdi'} ex) {mul_app_name: ["mulcli","mulfab","mull2sw","multr","mulmakdi"]}

10 Management Registration API

API	Task

Get {version}/regist	Show all gui call-back server regist in nbapi server. URL Input: - {version} : NBAPI Version. It is currently '1.0' Output: -ip : ip address of gui server -port : port number of gui server ex){gui_servers: ["10.1.100.140:1111"]}
POST {version}/regist/{port}	Regist gui call-back server to nbapi server. URL Input: - {version} : NBAPI Version. It is currently '1.0' - {port} : port number of gui callback server ex) {"gui callback server regiested : "ip:port":} Output : {"Regist gui callback server": "10.1.100.140:1111"}
Delete {version}/regist	Unregist all gui call-back servers from nbapi server with input ip. URL Input: - {version} : NBAPI Version. It is currently '1.0' Output : - success ex) {"gui callback server removed": [{"gui callback server": "10.1.100.140:1"}, {"gui callback server": "10.1.100.140:2"}, {"gui callback server": "10.1.100.140:3"}, {"gui callback server": "10.1.100.140:4"}]}