IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 24, 2019, accepted October 14, 2019, date of publication October 18, 2019, date of current version October 30, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2948215

Software-Defined MANET Swarm for Mobile Monitoring in Hydropower Plants

XI CHEN[1,2], TAO WU[3], (Member, IEEE), GANG SUN[2], (Member, IEEE), AND HONGFANG YU[2], (Member, IEEE)

[1]School of Computer Science and Technology, Southwest Minzu University, Chengdu 610041, China
[2]School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
[3]School of Computer Science, Chengdu University of Information Technology, Chengdu 610225, China

Corresponding author: Tao Wu (wut@cuit.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFC1507005, in part by the China Postdoctoral Science Foundation under Grant 2018M643448, in part by the Sichuan Science and Technology Program under Grant 2019YFG0110, and in part by the Fundamental Research Funds for the Central Universities, Southwest Minzu University, under Grant 2019NQN22.

ABSTRACT The construction and running of hydropower plants does not merely involve hydropower generation itself but also facilities surveillance, water quality monitoring, harmful creatures tracking, etc., thus requiring mobile and collaborative monitoring capabilities. MANET (Mobile Ad Hoc Networks), with its mobility, flexibility, and robustness in volatile networking environment, is a competitive candidate to fulfill such tasks. Nevertheless its distributed structure prevents the effective collaboration between MANET nodes. SDN (Software-Defined Networking) provides the centralized control over the network underlay. This paper proposes the SDN-controlled IEEE 802.11 MANET swarm for mobile monitoring in hydropower plants. First, the MANET node is implemented by integrating Raspberry Pi with cameras, various sensors, etc., into the low-cost wheeled mobile hardware to enable sensibility and mobility. Then, multiple such MANET nodes are networked through ad hoc protocols, to construct a flexible and distributed MANET underlay; meanwhile, to implement centralized control over the MANET underlay, every MANET node is equipped with OpenFlow switch software (e.g., Open vSwitch) so that OpenFlow directives issued by the SDN controller can be understood and executed, hence the SDN overlay on top of the MANET underlay. Finally, OpenFlow is extended to offer physical actions such as mobility and sensibility manipulation, beyond pure data forwarding in traditional SDN applications, to achieve mobile monitoring in hydropower plants. The SDN-controlled MANET swarm features mobile and sensing capabilities, flexible networking through ad hoc protocols, and efficient and unified control by SDN. Experiment results prove the feasibility of this network architecture.

INDEX TERMS Hydropower plant monitoring, mobile ad hoc network (MANET), software-defined networking (SDN), link layer discovery protocol (LLDP).

I. INTRODUCTION

Hydropower plays a key role in the energy industry. The construction and running of hydropower plants does not merely involve hydropower generation itself due to the complex environmental and hydrological conditions they reside in. It also requires monitoring over power generation facilities themselves to ensure energy safety, as well as the monitoring over hydrological status and water quality, rare species tracking to evaluate ecological impact, water level measurement for flood prevention, etc. Given such a complex and dynamic environment and diverse targets to monitor, simply introducing fixed sensors and cameras might not fully meet the monitoring requirements. For example in a hydropower plant as shown in Figure 1, power generation or auxiliary facilities might not be effectively monitored due to that their positions are out of the range of fixed sensors/cameras, or obscured by unexpected obstacles such as growing tree branches on the riverside (Q1 in Figure 1, i.e., an obscured voltage transformer). Facilities might also be damaged by wild animals (Q2), e.g., birds. Besides, sensors/cameras themselves might also need to be checked/monitored

The associate editor coordinating the review of this manuscript and approving it for publication was Lie-Liang Yang.

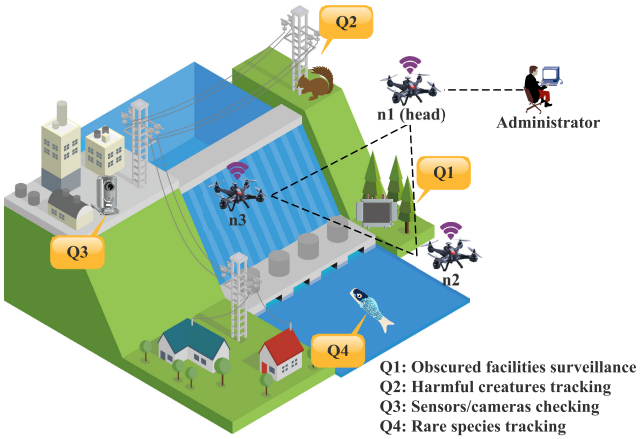


FIGURE 1. Hydropower plant monitoring requirements.

on a periodic basis or in an ad hoc fashion to guarantee they work properly (Q3). Also, the construction and running of a hydropower plant might lead to ecological impact so that some rare species should be tracked for environmental impact evaluation (Q4). In such scenarios, a swarm of mobile nodes (such as unmanned aerial vehicles n1, n2, and n3) can be notified to patrol the target spot and accordingly adapt on-site formations (for example the triangle formation in the figure) for a better monitoring angle, to avoid obstacles, to track harmful creatures, etc., and transfer surveillance or damage report to administrators through long-range low-power communications so that facility loss can be controlled or even avoided at early stage.

MANET (Mobile Ad Hoc Networks) [1], with its mobility, flexibility, and robustness in volatile networking environment, is a competitive candidate for such complex monitoring tasks. Nevertheless, the MANET swarm by nature is a distributed structure, hence the challenge in consistent and streamlined collaboration during mobile surveillance. To uniformly and coherently control a MANET swarm is non-trivial task. The administrator can choose to conduct remote node-wise control. But this approach does not scale for a larger number of MANET nodes. The administrator can also choose to conduct remote swarm-wise control where he/she directly controls only the ''head'' (n1 in Figure 1) of the MANET swarm, and the ''head'' controls other MANET nodes on behalf of the administrator, to achieve centralized management and controlled swarm collaboration. SDN (Software-Defined Networking) [2], which controls network devices in a centralized manner, is a competitive candidate for this purpose. Recent years have witnessed enormous applications of SDN in DCN (Data center Networks) [3]–[5], WAN (Wide Area Networks) [6], [7], network virtualization [8], [9], network resources optimization [10], [11], QoS provisioning [12]–[14], service function chaining [15]–[18], etc. The successful application of SDN in these fields lies within its centralized control over the network underlay. This idea can be borrowed and applied in MANET-based sensing and monitoring so as to control MANET devices consistently and effectively in a centralized manner. SDN separates control plane and data plane of network devices (switches, routers, etc.), so that complex control logic is abstracted away from these hardware devices and aggregated in the central softwarized SDN controller, leading to network devices with only simpler forwarding logic to achieve higher forwarding efficiency. The SDN controller watches over the entire network since the global topological view can be acquired and updated using continuous LLDP (Link Layer Discovery Protocol) [19] multicast. Network devices communicate with the centralized SDN controller using OpenFlow [20] so that they can be effectively controlled and the data forwarding can be directed by OpenFlow actions. OpenFlow and LLDP comprehensively lay the foundation of the control plane (i.e., the overlay) that manages the concrete network hardware (i.e., the underlay) so that vendor-specific details are encapsulated and hidden. While OpenFlow, LLDP, etc., work as the southbound interface (controller-device interface), the SDN controller also provides northbound interface (controller-application interface, such as REST API) to rapidly and uniformly develop network applications.

There are existing works in the literature that integrate SDN into distributed network underlays, which mainly focused on network issues such as security [21], [22], access control and flow scheduling [23], data aggregation [24], machine learning integration [25], etc., detailed in Section V. Nevertheless, the collaborative mobile monitoring based on the SDN-controlled MANET swarm in an infrastructure-free environment, as well as the direct control over MANET nodes through extended OpenFlow actions, is seldom studied. In this paper, we propose the SDN-controlled IEEE 802.11 MANET swarm, an approach to construct a centralized SDN overlay on top of the distributed MANET underlay for mobile monitoring in hydropower plants. The contribution of this paper includes as follows:

- VxLAN (Virtual eXtensiable Local Area Network) [26] tunnelling that bridges the ad hoc protocol stack and SDN protocol stack, so that the data forwarding of the MANET underlay can be uniformly and coherently controlled by the centralized SDN controller.
- LLDP is extended to offer metrics collection capabilities, so that various MANET node metrics such as battery, sensing features, etc., can be transparently collected during native topology discovery.
- OpenFlow is extended to offer physical actions so that sensing and mobility of the MANET underlay can be consistently controlled without involving hardware details.
- A prototype of low-cost and multifunctional MANET swarm controlled by SDN is proposed based on the Raspberry Pi platform. It features an SDN layer that enables overlay networking through best-practice SDN methodologies. Extensive experiments in a real-world hydropower plant were conducted to demonstrate the feasibility of the SDN-controlled MANET swarm.

The rest of this paper is organized as follows: Section II explains the motivations of this work. Section III specifies how the SDN overlay is built on top of the MANET underlay, including MANET swarm node design, the protocol tunnelling and bridging, enhanced topology discovery, centralized controlling, and the extension to OpenFlow actions. Section IV conducts various experiments to test the functionalities and performance of the proposed SDN-controlled MANET swarm. Related works are summarized in Section V. Finally, this paper is concluded, and future works are envisioned in Section VI.

II. MOTIVATIONS

Our work of integrating SDN over MANET for mobile monitoring is motivated by the following aspects.

- Collaborative monitoring and improved control: As we have explained in Figure 1, for a dynamic environment (e.g., a hydropower plant), it requires collaboration such as obstacles avoidance, on-site formation adaptation, etc., to conduct effective monitoring. This can be achieved by applying centralized control such as SDN on a swarm head to manage and coordinate the whole distributed MANET swarm. SDN on top of MANET brings easier manageability in addition to flexibility, robustness, etc., brought by MANET. Also, the remote administrator mainly communicates with the swarm head to conduct simpler swarm-wise control compared with distributed ad hoc control. Meanwhile, the ad hoc data forwarding can be optimized with centralized control introduced by SDN.
- Wired network infrastructure integration: The primary use of SDN in industry and literature is to enhance functionalities or performances of wired networks (e.g., TCP/IP) using centralized control. Given a mobile distributed wireless swarm, it would be much simpler to integrate the swarm into a deployed wired monitoring network if SDN technologies are applied in the swarm, since applications of SDN can already be found in monitoring facilities (see Section V). The integration and collaboration of wired and wireless monitoring networks also makes possible the 2-dimensional (i.e., fixed-mobile) monitoring in a complex environment.
- Unified interface and development: Data forwarding and physical monitoring/sensing involves network protocols and specific device drivers, respectively, which might result in a two-tier development for networked monitoring devices. If these two aspects can somehow be integrated, it could give a unified invocation interface and simplify device development through the SDN northbound interface and the extended OpenFlow that governs both data forwarding and physical actions (e.g., moving, sensing, etc.). Compared with simple REST invocations for the same purpose, the control-plane communication is natively protected by the secure SDN channel, which also reduces the security burden.

Based on these motivations, we studied the software-defined MANET swarm for mobile monitoring, designed and applied the prototype in a hydropower plant, and evaluated its functionalities and performance.

III. SOFTWARE-DEFINED MANET SWARM

A. MANET SWARM NODE

We construct the MANET node based on Raspberry Pi, an extensible multifunctional low-cost hardware platform. The architecture of the MANET swarm node is shown in Figure 2, which is roughly divided into 3 layers.

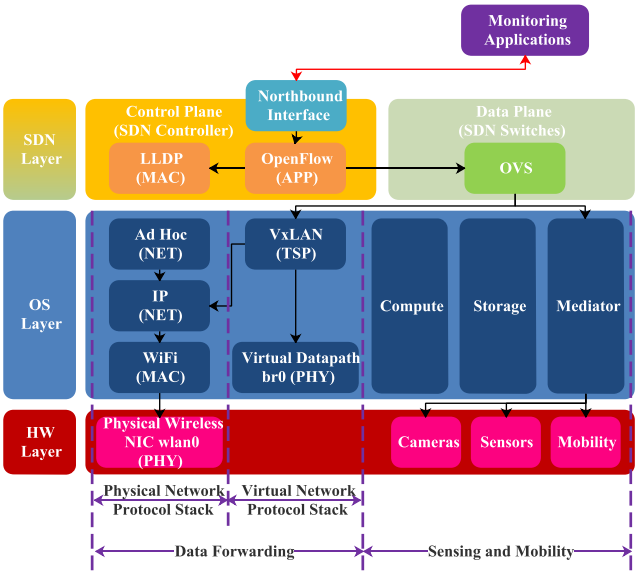


FIGURE 2. MANET swarm node architecture.

The HW (hardware) layer is based on the Raspberry Pi platform with wheeled mechanics, providing mobility, sensibility, camera surveillance, etc. This layer is also capable of various extension sensors through plug-in pins to strengthen sensing and monitoring. It provides various physical wireless communication devices as well. As shown in Figure. 2, there is an IEEE 802.11 (aka WiFi) NIC (network interface card) named wlan0 in our current design.

The OS (operating system) layer provides functionalities like networking, computing, storage, etc., as we can see from Figure 2. The native Raspberry Pi OS (i.e., Raspbian) as well as other Debian-spawned operating systems such as Ubuntu are preferred choices since SDN-related software to be exerted later for SDN overlay is compatible with these Debian derivatives. The OS layer must provide wireless protocols to pave the MANET communication foundation. Accepted choices include WiFi, ZigBee [27], LoRa [28], NB-IoT [29], etc. Since our MANET swarm might involve high data rate transmission in camera surveillance, we adopt WiFi as the fundamental networking mechanism between MANET swarm nodes. To guarantee WiFi communication and connectivity in a likely infrastructure-free environment for hydropower plant mobile monitoring, the OS layer should

also provide ad hoc routing protocols such as AODV [30], DSDV [31], DSR [32], OLSR [33], etc., for distributed data forwarding to construct the MANET underlay, detailed in Section III-B. In addition, virtualization capabilities are also needed in the OS layer to build an SDN overlay on top of the MANET underlay by means of SDN technologies. We can see from Figure 2 that we have virtualized a virtual datapath br0, to be later interfaced with virtualization technologies such as VxLAN, OVS, etc., for this purpose. Details of network virtualization and SDN overlay networking can be found in Section III-C.1. Besides, in order to uniformly control MANET sensing and mobility through SDN technologies in addition to pure data forwarding, a *mediator* module (see Figure 2) is designed to parse the extended physical actions (sensing, moving, etc.) contained in OpenFlow messages, and drive cameras, sensors, wheels, etc. Details of this mediator can be found in Section III-C.3.

The distinctive extension of our MANET swarm node is the SDN layer. Since MANET nodes work in an SDN-controlled distributed swarm under control of the swarm head, every node must be equipped with the SDN data plane, i.e., SDN switches such as Open vSwitch (OVS) [34]. Meanwhile, swarm head might be re-assigned to other MANET nodes due to damage, high workload, low battery, etc., hence every node must be eligible to be elected as the swarm head to take over the control over other MANET nodes. Thus, the SDN control plane, such as Floodlight, ONOS [35], OpenDaylight [36], etc., must be installed on every node as well. Therefore, the full SDN protocol stack, both data plane and control plane, must be deployed in addition to the network protocol stacks that come along with or manually installed on the Raspberry Pi platform. In this paper, we adopt OVS as the SDN switch since it is an easy-to-deploy lightweight software switch compatible with many open source systems such as Ubuntu. We also adopt the lightweight Floodlight as the SDN controller to reduce resource and battery consumption on the low-cost Raspberry Pi platform. SDN controller (i.e., Floodlight) is only activated on the MANET swarm head while SDN switch (i.e., OVS) is always standing by to accept controlling by the swarm head through OpenFlow. Details of the SDN overlay construction can be found in Section III-C.

B. MANET UNDERLAY NETWORKING

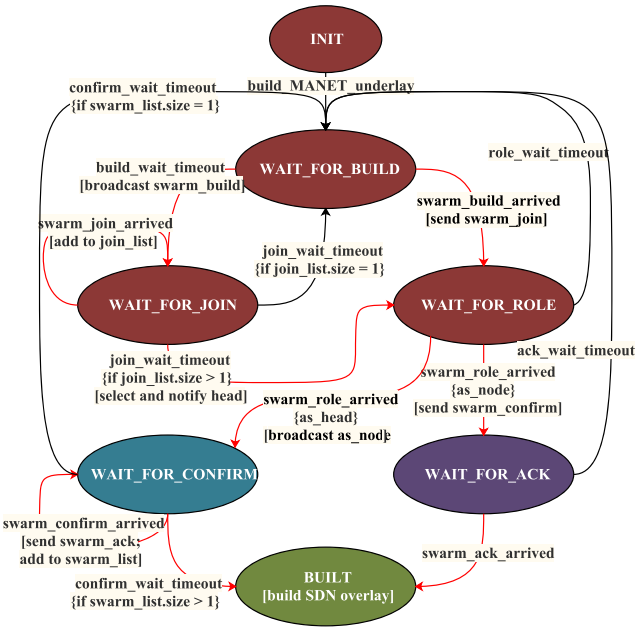
Every MANET swarm node needs to participate in a two-tier networking, i.e., the MANET underlay networking and the SDN overlay networking. Nevertheless, the MANET underlay does not have fixed networking infrastructures. Therefore, the physical wireless NIC wlan0 of the MANET swarm node should be configured to work in the ad hoc mode in the first place. Since many peer-to-peer networking protocols (e.g., AODV, DSDV, DSR, etc.) introduce CSMA/CA-like or RTS/CTS-like mechanisms similar to WiFi for MAC layer, we designed the WiFi-IP-Ad Hoc architecture for distributed MANET underlay networking, as shown in Figure 2. The second step is the IP configuration. All participating MANET nodes in the same swarm must be configured in the same IP subnet and the network mask can be tuned accordingly with regard to the scale of the swarm. Peer MANET swarm nodes can now ping each other using physical IP addresses and receive ICMP echo replies normally at this point of time, even though no APs (access points) are present.

After the previous configurations, the automatic MANET underlay networking process can be started in a peer-to-peer manner. We implemented the networking protocol, namely MSNP (MANET Swarm Networking Protocol), using Scapy, a Python library capable of sending, sniffing, dissecting and forging network packets. Scapy can also specify the workflow of self-defined network protocols by means of FSM (Finite State Machine). Figure 3a demonstrates MSNP's FSM, where ellipses represent states, and arrows present transitions. Arrow labels with {} indicate transition conditions and those with [] indicate operations. And Algorithm 1 demonstrates its workflow. MSNP resides in the network layer, and adopts IP as its underlying protocol. The *protocol* field of IP is unambiguously set to 0xFE (i.e., 254 in decimal) when taking MSNP as its payload. MSNP is a lightweight protocol with several simple-structured PDUs (Protocol Data Units), including *swarm_build*, *swarm_join*, *swarm_role*, *swarm_confirm*, *swarm_ack*, etc. There are three different roles during the networking process, namely *peer*, *head* and *node*, wherein the *head* controls *node*s in the SDN overlay to be constructed above the MANET swarm later in Section III-C.

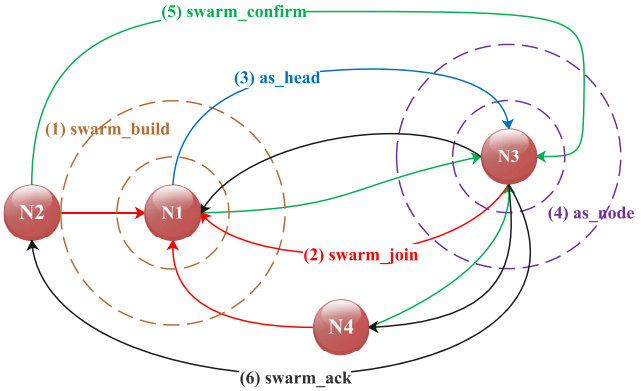
Every MANET node starts the networking as a *peer* and enters the WAIT_FOR_BUILD state with a timer of B seconds before it can initiate networking request (i.e., the *swarm_build* message), to avoid request collisions. During a period of B seconds, *peer*s wait for any possible incoming networking requests (i.e., the *swarm_build* message on line 6 of Algorithm 1). Once the B timer times out, a *peer* is free to broadcast a *swarm_build* so that other *peer*s can choose to jointly build a MANET swarm. Any *peer* that receives *swarm_build* sends back the *swarm_join* message if willing to join the swarm (line 7), together with its key metrics for wireless communications, such as battery information, for the following swarm *head* selection procedure, and enters the WAIT_FOR_ROLE state with a timer of R seconds to avoid dead lock. The initiating *peer* then enters the WAIT_FOR_JOIN state with a timer of J seconds, to continuously receive any possible *swarm_join*s, considering communication and processing delays. Once the J timer times out, it picks the best one (e.g., with the highest battery) among all *peer*s (including itself) and notifies it to be selected as the *head* (i.e., the as_head message on line 26), and enters the WAIT_FOR_ROLE state as well, to be later assigned a proper *swarm_role* by the *head*. Note that the initiating *peer* of the *swarm_build* may not necessarily become the swarm head, according to the protocol workflow. This gives a chance that the "best" peer can be fairly and optimally selected. Once notified with as_head, the *peer* sets itself as the swarm head and in turn notifies all other participating *peer*s of its role as the head and their roles

Algorithm 1 MSNP (MANET Swarm Networking Protocol)

```
1:  swarm_list = {MY_IP}; join_list = {MY_IP}
2:  role = "peer"; head = null
3:  enter WAIT_FOR_BUILD(timeout=B)
4:
5:  state WAIT_FOR_BUILD(timeout=B):
6:  if receive swarm_build then
7:      reply swarm_join
8:      enter WAIT_FOR_ROLE(timeout=R)
9:  else if timeout then
10:     broadcast swarm_build
11:     enter WAIT_FOR_JOIN(timeout=J)
12: end if
13:
14: state WAIT_FOR_JOIN(timeout=J):
15: if receive swarm_join then
16:     add to join_list
17:     enter WAIT_FOR_JOIN(timeout=J)
18: else if timeout and join_list.size == 1 then
19:     enter WAIT_FOR_BUILD(timeout=B)
20: else if timeout and join_list.size > 1 then
21:     send as_head to join_list.best
22:     enter WAIT_FOR_ROLE(timeout=R)
23: end if
24:
25: state WAIT_FOR_ROLE(timeout=R):
26: if receive as_head then
27:     role = "head"
28:     broadcast as_node
29:     enter WAIT_FOR_CONFIRM(timeout=C)
30: else if receive as_node then
31:     role = "node"
32:     head = sender of as_node
33:     reply swarm_confirm
34:     enter WAIT_FOR_ACK(timeout=A)
35: else if timeout then
36:     enter WAIT_FOR_BUILD(timeout=B)
37: end if
38:
39: state WAIT_FOR_CONFIRM(timeout=C):
40: if receive swarm_confirm then
41:     reply swarm_ack
42:     add to swarm_list
43:     enter WAIT_FOR_CONFIRM(timeout=C)
44: else if timeout and swarm_list.size == 1 then
45:     enter WAIT_FOR_BUILD(timeout=B)
46: else if timeout and swarm_list.size > 1 then
47:     enter BUILT()
48: end if
49:
50: state WAIT_FOR_ACK(timeout=C):
51: if receive swarm_ack then
52:     enter BUILT()
53: else if timeout then
54:     enter WAIT_FOR_BUILD(timeout=B)
55: end if
```



(a) Finite State Machine of MANET Underlay Networking



(b) A MANET Underlay Networking Example

FIGURE 3. MANET underlay networking.

as controlled swarm *node*s (i.e., the as_node message on line 28). Then, the *head* enters the WAIT_FOR_CONFIRM state with a timer of C seconds. Other *node*s reply with the *swarm_confirm* (line 40) and enters the WAIT_FOR_ACK state with a timer of A seconds. And the *head* acknowledges *swarm_confirm* with *swarm_ack*. Both the *head* and *nodes* enter the BUILT state where the building of the SDN overlay can be later initiated. The red transitions in Figure 3a indicate the successful networking of a MANET swarm while the timeout mechanism (black transitions) prevents possible dead lock due to communication delay, packet-drops, communication peer failures, etc.

Figure 3b gives an example on how MANET underlay networking is conducted. (1) N1 initiates networking request *swarm_build* (brown lines). (2) N2-N4 replies with *swarm_join* containing battery information (red arrows). (3) N1 selects N3 as the swarm head by comparing battery information and notifies N3 the result (blue arrows). (4) N3 sets itself into swarm head mode, and notifies all other

nodes of the swarm head selection result and their roles as controlled swarm nodes (purple lines). (5) All other nodes confirm to join the swarm (green arrows). (6) N3 acknowledges (black arrows) and the MANET swarm is built successfully.

C. SDN OVERLAY CONTROLLING

The design and work presented in Section III-B successfully constructs the MANET underlay. Notice that at this point of time, the MANET underlay is still purely a distributed structure where nodes are self-managed, including self-determined routing (i.e., ad hoc routing), and self-controlled mobility and sensibility. In this section, we construct an SDN overlay on top of the MANET underlay to build a MANET swarm whose data forwarding and physical actions are uniformly controlled by the SDN controller.

1) VXLAN-BASED TUNNELLING

The key to deploy the SDN overlay above the MANET underlay to offer consistent and streamlined control is the deployment of OVS, i.e., the softwarized SDN switch. When OVS receives traffic, the actual forwarding is done by datapaths, e.g., br0 in Figure 4. Datapaths can be created on demand by OVS, and equipped with multiple vports (virtual ports on a datapath) for traffic ingress and egress. When directing packets to another vport upon traffic arrival, a vport matches packet fields with OpenFlow-installed flow table entries, and forwards matched packets or inquire the SDN controller for miss-matched ones. The primary workflow of OVS is shown by Flow 1 and Flow 2 (i.e., the scarlet arrows) in Figure 4.

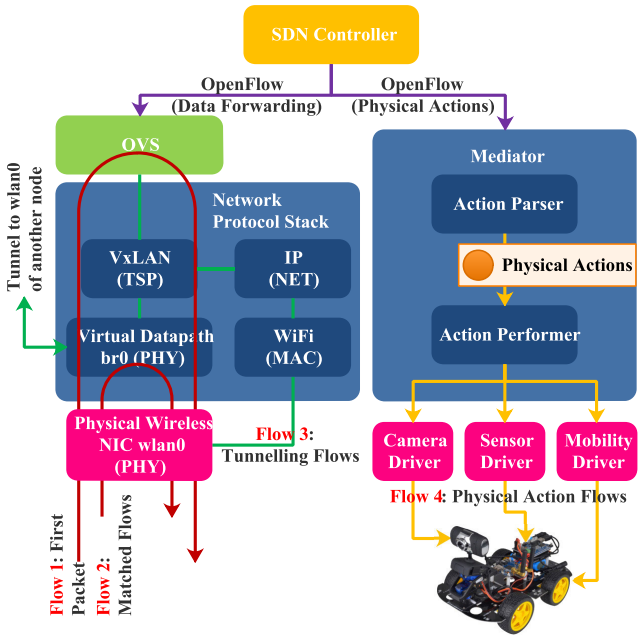


FIGURE 4. MANET swarm node workflows.

According to the workflow of the standard OVS mentioned above, to control the traffic flows running through the MANET underlay, wireless NICs of MANET swarm nodes must be bound to datapaths created by OVS, so that traffic flows run into OVS modules, leading to a direct control by OpenFlow. Nevertheless, the physical wireless NICs have already participated in the MANET underlay networking in the previous phase, thus simply bounding wireless NICs to datapaths fails the MANET underlay communication. Alternatively, we can create extra datapaths to encapsulate physical wireless NICs, so that the inner physical wireless NICs encapsulated by datapaths are controlled indirectly by OpenFlow. In this way, the SDN overlay becomes possible. On every MANET swarm node, we create an OVS datapath named br0 as shown in Figure 2. At this point of time, every node has two forwarding devices, the physical wireless NIC wlan0, participating in the MANET underlay networking, and the virtual datapath br0, to participate in the SDN overlay controlling. To encapsulate wlan0 without compromising the MANET underlay communication, it requires end-to-end tunnelling over the physical communication channels. To achieve this, VxLAN is adopted for pairwise tunnelling. VxLAN is a transport layer protocol of the OSI reference model, which transmits MAC layer frames using UDP protocol to enlarge the number of supported virtual networks. The ovs-vsctl utility provided by OVS offers VxLAN support. For every other MANET swarm node, br0 adds a vport to tunnel all the way to the wlan0 of the other node. Besides, br0 is assigned an independent virtual IP address ($IP_{virtual}$, e.g., 20.0.0.y) other than the physical IP address of wlan0 ($IP_{physical}$, e.g., 10.0.0.x). Flow 3 (i.e., the green lines) in Figure 4 shows the encapsulation process that tunnels physical wlan0 and virtual br0. Algorithm 2 shows the VxLAN tunnelling details.

Algorithm 2 VxLAN Tunelling

1: ovs-vsctl del-br br0
2: ovs-vsctl add-br br0
3: **for** every other *node* in *swarm* **do**
4: ovs-vsctl add-port br0 *vport*
5: ovs-vsctl set interface *vport* type=vxlan options: remote_ip= $node.IP_{physical}$
6: **end for**
7: ifconfig br0 $IP_{virtual}$ up
8: ovs-vsctl set-controller br0 tcp:$head.IP_{physical}$:6653

The ovs-vsctl utility also configures the SDN controller that controls OVS datapaths, i.e., line 8 in Algorithm 2. Given the fact that OVS is controlled by the SDN controller while virtual datapath br0 tunnels physical wlan0, it suggests that physical MANET swarm nodes can be indirectly controlled by the centralized SDN controller using VxLAN-based tunnelling, hence the SDN overlay on top of the MANET underlay.

2) LLDP-BASED TOPOLOGY DISCOVERY

Upon the successful construction of the SDN overlay above the MANET underlay, the SDN controller starts to watch over

the life cycle of the SDN overlay. The very first step is the topology discovery so as to gain a global overview of the network for reliable control. However, the SDN controller knows only the switches (i.e., the MANET swarm nodes with OVS deployed) directly connected to it. How these switches are mutually connected is unknown. MANET underlay utilizes wireless communication channels, which have the broadcasting feature, for data-links. Therefore, the MANET underlay topology presents a full mesh structure if nodes are close enough to each other. Nevertheless, the ''hidden terminal problem'' occurs when MANET swarm nodes are out of the wireless signal coverage of each other, yet somehow within the radio range of an intermediate node, say the swarm head with the SDN controller activated. In this situation, data-links between out-of-coverage nodes are disconnected whereas the data-links with swarm head are still connected, breaking down the full mesh topology. Therefore, the topology discovery is quite necessary for the swarm head to grasp an accurate global topological view.

Topology discovery and its continuous update in SDN is usually conducted using LLDP. Packet-out PDUs (OpenFlow messages from the SDN controller to switches to deliver instructions) containing LLDP are periodically initiated by the SDN controller. Switches receiving these packets (i.e., the senders) are instructed to multicast the LLDP PDU to all other connected switches (i.e., the receivers) by OpenFlow actions contained in packet-out. Receivers miss-match these LLDP packets due to the non-existence of proper flow table entries capable of LLDP PDU processing, leading to the packet-in inquiries from receiver switches to the SDN controller. These packet-ins contain link layer information of both senders and receivers, therefore, the SDN controller can safely assert the existence of data-links between senders and receivers. In this way, the SDN controller draws the global topology view and its subsequent updates during the life cycle of the SDN network.

LLDP features its extensibility enabled by TLV (type/length/value) fields. Self-defined information can be contained in TLVs. The MANET underlay is a distributed peer-to-peer network that works in an ad hoc fashion. The topology changes over time, and so does the battery, which is a critical factor for the election of MANET swarm head. Meanwhile, other metrics such as processing capability, bandwidth, sensory functions (e.g., cameras, temperature sensors) that the node provides, etc., also affect how nodes collaborate with each other as well as the overall planning and optimization by the centralized SDN controller. Should this extra information be included in TLVs of LLDP PDUs, various sophisticated managements can be implemented during topology discovery, such as planning the best transmission path that consists of nodes with most battery. Therefore, we extend the original LLDP with self-defined TLVs, as shown in Figure 5. Note that for different applications, different TLVs and the length for each TLV can be derived other than those in Figure 5. It is called the Metrics over LLDP (MetLLDP) in this paper to indicate that LLDP is adopted as the ferry that collects various

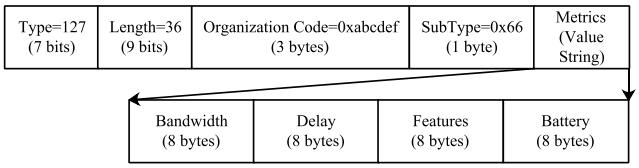


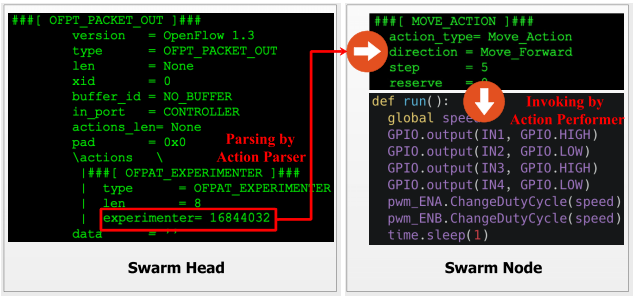
FIGURE 5. The MetLLDP PDU.

metrics of underlying MANET swarm nodes. Performance evaluation of MetLLDP for SDN overlay topology discovery will be conducted in Section IV-C.

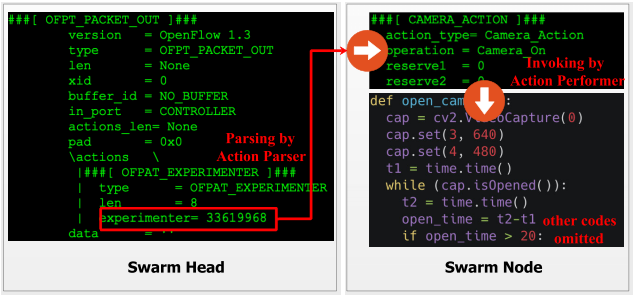
3) OPENFLOW-BASED CONTROLLING

Since swarm nodes involve moving, sensing, monitoring, etc., to fully implement the control and management of swarm nodes by means of SDN, OpenFlow must be extended with corresponding physical actions. According to the OpenFlow specification, the OFPAT_EXPERIMENTER construct can be used to incorporate self-defined or experimental actions. An OFPAT_EXPERIMENTER contains three fields, namely, *type* (1 byte), *len* (1 byte), and *experimenter* (4 bytes encoded in integer format). We made good use of the 4-byte *experimenter* field to derive more than 10 actions for the control of mobility, sensibility, surveillance, roughly divided into two types, i.e., MOVE_ACTION (e.g., Move_Forward, Move_Left, etc.) and SENSE_ACTION (e.g., Camera_On, Temperature_Humidity, etc.) for the time being. The first byte *action_type* in the *experimenter* field is used to indicate the type of physical actions, MOVE_ACTION or SENSE_ACTION. The following bytes are type-dependent fields. For example, the second field *direction* for MOVE_ACTION contains several enumerated values such as Move_Forward, Move_Backward, etc. These actions are encapsulated inside packet-out PDUs, sent from the swarm head to instruct swarm nodes. Note that the *experimenter* field is encoded in integer format upon sending, according to the specification. For example, when sending a MOVE_ACTION (*action_type* = 0x01) that instructs Move_Forward (*direction* = 0x01) by 5 units (*step* = 0x05), the *experimenter*, whose byte presentation is 0x01010500 (with a trailing *reserve* field 0x00), will be encoded as 16844032 in decimal integer, as shown in Figure 6a.

These actions are parsed back to byte presentation by the Action Parser component of the recipient swarm node upon packet-out arrival, so as to identify physical actions, as shown in Figure 6. And subsequently, hardware driver APIs are invoked internally to drive corresponding equipments on these swarm nodes. These operations are conducted by the mediator module as shown in Figure 4. Two approaches can be adopted to implement the mediator module. (1) Direct integration into OVS since OVS is the core entity to handle OpenFlow actions. It might be the most ''orthodox'' approach with regard to SDN methodologies. However, this approach requires deep dive into the OVS architecture and heavy development and integration efforts. Also, it might compromise ordinary data forwarding inside OVS if bugs arise during



(a) Parsing and Performing Move_Forward



(b) Parsing and Performing Camera_On

FIGURE 6. Parsing and performing physical actions from PACKET_OUT.

code modification. The last but not the least, the tight coupling between the mediator module and OVS might hinder the extensibility of physical actions in long term. (2) The non-intrusive modular approach. When OpenFlow messages containing extended physical actions beyond pure data forwarding actions arrive, the mediator module still parses and performs these physical actions, but it independently works in userspace and bypasses the OVS stack, without any OVS modules involved. This approach gives better modularity and freedom for extensibility. Following the non-intrusive approach, we implemented the bypass mediator module using Scapy. Upon the arrival of an OpenFlow message, the Scapy sniffing procedure determines if it contains any physical actions. The Action Parser component of the mediator, implemented as a Scapy FSM state, is notified upon the presence of physical actions and extracts them from the OpenFlow OFPAT_EXPERIMENTER construct. The extracted physical actions are then passed to the Action Performer component (also a Scapy FSM state) which drives various equipments on the swarm node by invoking corresponding drivers.

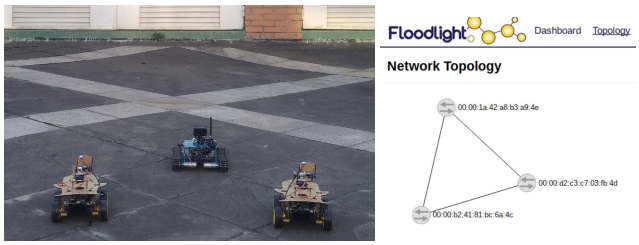
GPIO (General-Purpose Input/Output) is adopted as the core driver mechanism for Action Performer. Any of the GPIO pins can be designated (in software) as an input or output pin and used for a wide range of purposes. With the help of PWM (pulse-width modulation), various complex actions, such as moving, spinning, temperature/humidity sensing, etc., are implemented. Also, OpenCV is used to control the surveillance function. Figure 6a gives an example on how Move_Forward physical action is interpreted and then performed by the corresponding GPIO procedure *run*(). Figure 6b gives an example on how Camera_On physical action is interpreted and then performed by the corresponding OpenCV procedure *open_camera*(). Verbosity on GPIO/OpenCV and other examples are omitted due to paper length.

IV. EVALUATION

In this section, we implement the SDN-augmented MANET swarm node platform, construct the SDN-controlled MANET swarm and conduct various experiments to test functionalities and performances. The prototype of the MANET swarm is deployed in Shuanghe Hydropower Plant located in Dujiangyan, Sichuan Province, China, for these tests.

A. SDN-CONTROLLED MANET SWARM NETWORKING

The prototype of the SDN-controlled MANET swarm node is implemented based on the Raspberry Pi platform, augmented with wheels, cameras, sensors, etc., that is functioning as the HW layer as specified in Figure 2. Raspbian operating system is installed on every Raspberry Pi to construct the OS layer. Floodlight is deployed as the SDN controller, and OVS is also installed as the SDN data plane, to comprehensively build the SDN layer over every MANET swarm node. MetLLDP and OpenFlow are used for topology discovery with metrics collection and swarm control, respectively. In this experiment, without losing generality, three such MANET swarm nodes (P1-P3) are participating in the networking, among which, one node is working as the swarm head, as well as an ordinary MANET swarm node under the control of itself. Others automatically join the swarm by the mechanism specified in Section III-B, III-C.1, etc., as shown in Figure 7a. Then the swarm head conducts the topology discovery with metrics collection based on MetLLDP specified in Section III-C.2, as shown in Figure 7b. Details of the performance of MetLLDP can be later found in Section IV-C.



(a) Physical MANET Swarm (b) Logical SDN Overlay Topology

FIGURE 7. SDN-controlled MANET swarm.

The SDN overlay works in the 20.0.0.0/24 IP segment while the MANET underlay works in the 10.0.0.0/24 IP segment. We first check the connectivity of the SDN-controlled MANET swarm by means of pairwise pinging. Upon the successful construction of the SDN overlay, pairwise pinging is automatically enabled since by default the L2 learning function of Floodlight is enabled. Administrators can push flow entries through northbound REST API (relative URL is /wm/staticflowpusher/json) to instruct data forwarding, demonstrated by Figure 8. A flow entry that drops all data

```
pi@raspberrypi:~ $ curl -X POST -d '{"switch":"00:00:d2:a4:ec:11:6c:42","name":"
flow-1","in_port":"1","src-ip":"20.0.0.1","dst-ip":"20.0.0.4","active":"true"}'
http://10.0.0.4:8080/wm/staticflowpusher/json
{"status" : "Entry pushed"}pi@raspberrypi:~ $
```

(a) The flow entry that drops traffic between P1 and P3

```
pi@raspberrypi:~ $ ping 20.0.0.4 -c 4
PING 20.0.0.4 (20.0.0.4) 56(84) bytes of data.
From 20.0.0.1 icmp_seq=1 Destination Host Unreachable
From 20.0.0.1 icmp_seq=2 Destination Host Unreachable
From 20.0.0.1 icmp_seq=3 Destination Host Unreachable
From 20.0.0.1 icmp_seq=4 Destination Host Unreachable

--- 20.0.0.4 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3089ms
pipe 4
```

(b) Traffic disabled

```
pi@raspberrypi:~ $ curl -X DELETE -d '{"name":"flow-1"}' http://10.0.0.4:8080/wm
/staticflowpusher/json
{"status" : "Entry flow-1 deleted"}pi@raspberrypi:~ $
```

(c) Deleting the traffic-dropping flow entry

```
pi@raspberrypi:~ $ ping 20.0.0.4 -c 4
PING 20.0.0.4 (20.0.0.4) 56(84) bytes of data.
64 bytes from 20.0.0.4: icmp_seq=1 ttl=64 time=13.2 ms
64 bytes from 20.0.0.4: icmp_seq=2 ttl=64 time=3.31 ms
64 bytes from 20.0.0.4: icmp_seq=3 ttl=64 time=5.41 ms
64 bytes from 20.0.0.4: icmp_seq=4 ttl=64 time=1.99 ms

--- 20.0.0.4 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 1.999/5.982/13.200/4.342 ms
```

(d) Traffic enabled

FIGURE 8. Flow entries pushing.

from source IP 20.0.0.1 (i.e., P1) to destination IP 20.0.0.4 (i.e., P3) in this experiment is pushed by the administrator, as shown in Figure 8a. We can see from Figure 8b that subsequent pingings complain about the ''destination host unreachable'' (a type of ICMP error messages). Dynamic controlling over data forwarding of the MANET underlay is also implemented, as shown in Figure 8c where a flow entry that deletes the previous traffic-dropping instruction is pushed by the administrator. We can see from Figure 8d, the pinging is re-activated. Notice that the pinging operation directly uses the SDN overlay IP addresses (20.0.0.x), instead of those of the MANET underlay (10.0.0.x). This indicates that the forwarding of the MANET underlay is fully controlled by the SDN overlay, without any underlying hardware/configuration/communication details involved. Therefore, the administrator interacts with only the MANET swarm head to implement full control over the whole MANET swarm through the SDN layer deployed on every node.

We also tested the coverage of the SDN-controlled MANET swarm. For the typical open-air environment in the hydropower plant, it promises an efficient node-to-node radio coverage as long as approximately 55 meters. This coverage guarantees the effective communication and collaboration for typical mobile monitoring in hydropower plants.

B. SDN-CONTROLLED MANET SWARM MOBILE MONITORING

SDN-controlled MANET swarm mobile monitoring is conducted in this section. The monitored target is a voltage transformer facility, as shown in Figure 9a. This facility is

(a) The Monitoring Target – a Voltage Transformer Facility

(b) Fixed Camera Monitoring

(c) MANET Swarm Mobile Monitoring

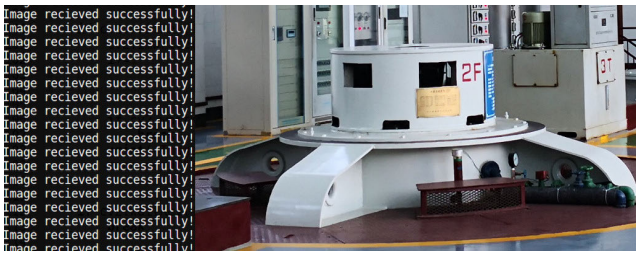
FIGURE 9. SDN-controlled MANET swarm mobile monitoring.

obscured by nearby tree branches at large, thus hard to be effectively monitored by the fixed camera, as we can from Figure 9b. We deployed a MANET swarm consisting of three wheeled mobile nodes. The swarm head send OpenFlow messages that contain the two types of physical actions mentioned in Section III-C.3: MOVE_ACTION and SENSE_ACTION. A flow entry containing action Camera_On is pushed by the SDN controller, and the MANET swarm node that receives it turns on its camera to conduct live surveillance. To bypass obstacles and get better monitoring angles, the swarm head also instructs swarm nodes to gradually move around the monitored target while conducting live surveillance, by sending OpenFlow messages that contain movement actions in different directions. Figure 9c shows this mobile monitoring process that gradually revealed the fore-side and left-side of the voltage transformer facility. These physical actions are particularly useful for collaborative mobile surveillance over valuable assets or facilities. Combination of movement actions would effectively adapt on-site formation to gain a better surveillance angle, track harmful moving objects, etc.

The MANET swarm can also conduct a multi-angle and closer monitoring over the indoor hydropower generators, as shown in Figure 10a. This is quite useful for automatic preliminary diagnoses in case of facility failures or malfunctions. For example, the swarm can move around the generator to preliminarily check if there is any rats-bitten wire or too much rust that prevents the generator from properly functioning, before staff attendance. The swarm can also conduct ad-hoc/on-demand patrols or facility checks for areas that are not equipped with cameras or sensors. Figure 10b shows the live surveillance by the MANET swarm over the discharge dam that is out of the coverage of fixed cameras.

C. SDN OVERLAY TOPOLOGY DISCOVERY

In this section, MetLLDP is tested for its capabilities of acquiring topological information as well as various metrics.



(a) Indoor Hydropower Generator Monitoring

(b) Outdoor Water Flow Monitoring

FIGURE 10. SDN-controlled MANET indoor/outdoor monitoring.

Two different topologies are deployed for the test, a mesh topology with 9 participating MANET swarm nodes, and a topology with 4 MANET swarms, each with 5 nodes. The experiment purpose is to test the performance of topology discovery capabilities in term of traffic overhead resulted by topology discovery with metrics collection. This experiment is conducted in the above topologies under different traffic settings (with or without camera surveillance, i.e., Camera_On v.s. Camera_Off). The experiment counterparts are the proposed MetLLDP, sFlow [37] and the original LLDP without metrics collection extension.

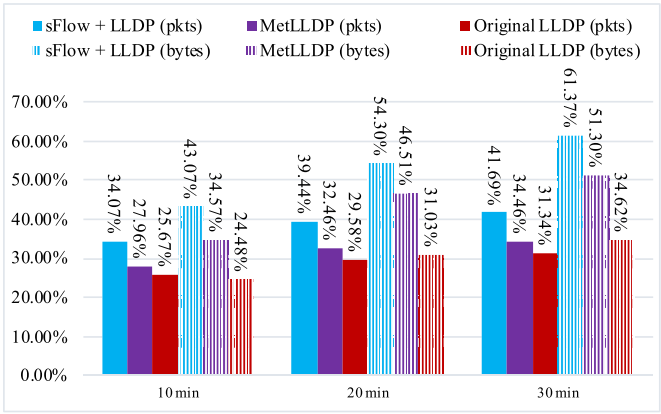
Counterpart sFlow is a widely used network monitoring system. However, the topological information acquisition is inaccurate and its topology discovery does not work well in SDN environments compared with the SDN-native LLDP topology discovery [38]. Therefore, if sFlow is to be used for topology discovery in addition to the metrics collection capabilities, it must work with the original LLDP. That is to say, both protocols must be deployed, which consequently adds extra deployment efforts as well as administrative burden. On the other hand, the metrics of MANET swarm nodes collected by sFlow and topological information collected by the original LLDP must be carefully aggregated to generate an accurate and consistent metrics-aware topology view, since these two types of information do not arrive at the MANET swarm head synchronously. Instead, sFlow sampling interval might be different with that of the original LLDP in the first place. Even if the intervals are configured exactly the same, metrics and topology information collection operations might be initiated at different times, requiring synchronization upon arrivals. Therefore, sFlow based metrics collection is a challenging task that requires plentiful programming efforts. The misplacement of metrics collected by sFlow on MANET swarm nodes discovered by LLDP would cause serious bad decision-making for metrics-aware applications [38]. Another problem is, when deploying two protocols (original LLDP and sFlow), the faster battery and computational resource consumption, which is a critical factor for wireless ad hoc environment.

MetLLDP, on the other hand, is an extended protocol based on LLDP that contains self-defined TLVs for metrics collection. It natively inherits topology discovery capabilities from LLDP. In addition, it collects metrics the same time when topology discovery happens, hence the native synchronization between these two types of information. Therefore, no extra efforts are need to synchronously aggregate topological information and the metrics information. MetLLDP functionally outperforms sFlow with regard to information synchronization.

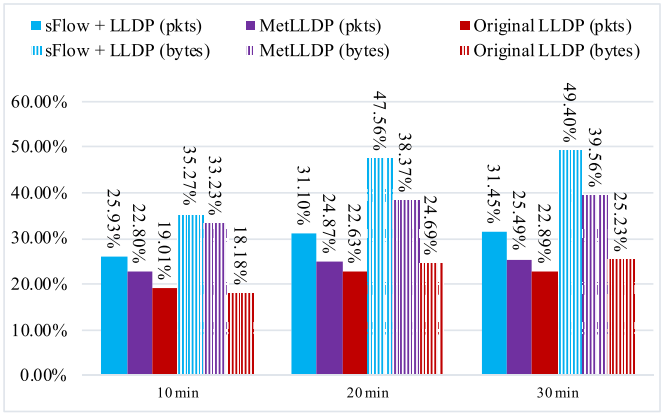
Performance experiments are also conducted where sFlow and MetLLDP are compared in terms of runtime traffic overhead during topological and metrics information collecting under different topologies. Three different schemes are tested in this section, the original LLDP for pure topology discovery, the sFlow + LLDP scheme for both topological and metrics collection, and MetLLDP for both topological and metrics collection. Sampling intervals of MetLLDP and sFlow are set the same as the default sampling interval of the original LLDP, i.e., 15 seconds, for fair comparison.

The experiment results are shown in Figure 11. Red columns exhibit the traffic overhead introduced by the original LLDP during topology discovery and continuous update over time, where solid columns indicate the overhead ratio to the full traffic in terms of packets and striped columns indicate the overhead ratio to the full traffic in terms of bytes. Purple columns (both solid and striped) indicate traffic overhead by MetLLDP while blue columns indicate that by sFlow + LLDP. Experiment statistics are recorded for 10, 20 and 30 minutes. Figure 11a and 11b are experiment results without camera surveillance and live video traffic transfer while Figure 11c and 11d indicate results with live camera surveillance and video streaming from one MANET swarm node to another one. In this experiment, Wireshark is used for traffic capture and analysis.

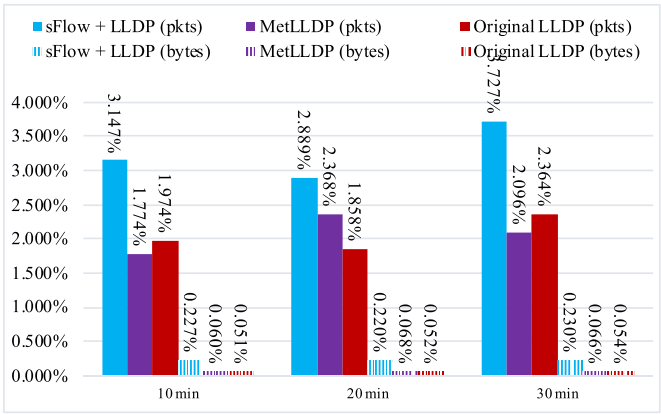
The traffic overhead introduced by the original LLDP is treated as the benchmark to evaluate the performance of topology discovery with metrics collection by the other two schemes, i.e., MetLLDP and sFlow + LLDP. According to the experiment results, traffic overhead caused by MetLLDP is slightly greater compared with the original LLDP benchmark. Take Figure 11d 20 min as an example, MetLLDP traffic overhead in bytes is 0.027% while the original LLDP benchmark overhead is 0.021%, indicating metrics collection during topology discovery at very low traffic cost. However, sFlow + LLDP scheme produces much greater traffic overhead compared with the original LLDP benchmark, about 7.9 times as much, due to much larger packet encapsulation (up to 592 bytes) by sFlow for metrics collection. Similar analysis applies to Figure 11a–11d. To summarize, MetLLDP provides desirable metrics collection performance along with excellent native topology discovery capabilities, at the cost of very low traffic overhead due to its piggyback methodology.



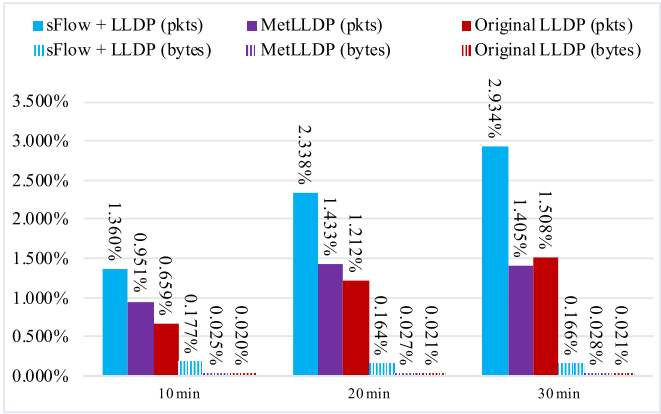
(a) Traffic Overhead under Mesh Topology without Camera Surveillance



(b) Traffic Overhead under Multiple-Swarm Topology without Camera Surveillance



(c) Traffic Overhead under Mesh Topology with Camera Surveillance



(d) Traffic Overhead under Multiple-Swarm Topology with Camera Surveillance

FIGURE 11. **Traffic overhead comparison: original LLDP, sFlow + LLDP, and MetLLDP.**

This is especially important for battery consumption reduction in wireless environment. Meanwhile, given its concise and compact packet encapsulation, it is also beneficial to bandwidth saving for wireless, especially narrow-band, communications.

V. RELATED WORKS

1) SDN INTEGRATION INTO WIRELESS NETWORKS

The SDN paradigm has been extended from wired networking towards the wireless networking after years of research and development. There have been attempts of applying SDN in wireless networking since earlier researches. OpenRoads [39] merged SDN with wireless networking technology, especially with WiFi and WiMAX technologies, and deployed it in the Stanford University campus network. It can be regarded as the wireless version of OpenFlow. The architecture of OpenRoads consists of three layers, including the flow table layer, which is jointly coordinated by the OpenFlow and WiFi/WiMAX; the slicing layer, where the network is sliced using FlowVisor; and the controller layer, which is implemented by NOX [40]. Users can conduct experiments on top of the controller layer by simply invoking the northbound interfaces of SDN without diving deep into the details of wireless protocols. In particular, OpenRoads demonstrated how wireless handover can be effectively conducted by means of SDN in wireless environments. OpenRadio separates a wireless network into a programmable data plane and a declarative decision plane. Network operators need only declare the rules of the decision plane, which are parsed and enforced by the data plane, to implement the effective management and policy-specific routing. In reference [41], OpenFlow is applied to WMN (Wireless Mesh Network), and an SDN solution based on virtualization technology and cross-layer flow table rules for WMN is proposed to enable unified control. Similarly, applications of SDN in wireless networks are also reported in references [42], [43]. These works focused on the study of the SDN-equipped wireless networks, lacking the support for ad hoc networking. Meanwhile, physical actions of mobile nodes are not uniformly controlled by OpenFlow as done in our work.

2) SDN INTEGRATION INTO MANET

Reference [44] proposed to combine SDN and MANET in the networking and collaboration of smart phones.

The distributed AODV routing protocol is used for the basic peer-to-peer connectivity. Each device obtains the global network view by scanning. Meanwhile, each node is a device with controller and switch capabilities, capable of controlling or being controlled by other nodes, thus an ad hoc architecture. However, standard SDN protocols, such as OpenFlow, LLDP for link discovery, etc., are not used in this scheme. Therefore, the interconnectivity between the proposed scheme and traditional SDN (such as wired SDN) is hard to achieve, thus limited application value in scenarios where wired sink nodes/gateways are required. Our work differs from this work in that de facto SDN protocols such as OpenFlow, LLDP, etc., are used in the proposed scheme hence the interoperability between the proposed SDN-controlled MANET swarm and the wired SDN backbone. Reference [45] proposed and envisioned the software-defined VANET architecture and various services. Architecturally, it includes a remote centralized SDN controller (usually located in a telecommunications data center), base stations, fixed RSUs (Road Side Unit), and mobile vehicles. Vehicles and RSUs constitute the data plane, whose data transmission is controlled by the controller. Base stations and the remote controller conduct remote communication via wireless protocols, such as LTE or WiMAX. This work gave some preliminary envisions on the software-defined VANET but very little implementation. Our work differs from this work in that SDN practice is conducted on top of the MANET underlay, offering meaningful and unified control over the data forwarding, sensing, mobility, etc., of the MANET swarm. Reference [46] proposed to integrate SDN into MANET, so that distributed MANET can be controlled by the centralized SDN controller at runtime. This work presented architecture design and implementation details. The key idea was to control the OVS entities deployed on MANET nodes in a centralized manner by the SDN controller. Best practices of SDN, such as LLDP-based discovery, OpenFlow-based control, were applied for this purpose, thus basic interoperability with other SDN networks was maintained. It also claimed that the proposed scheme outperforms the pure MANET that routes using OLSR (Optimized Link State Routing) protocol in terms of throughput in simple topologies. The significance of this work lies within the practical bridging between the SDN protocol stack and MANET protocol stack, which proved the feasibility of the idea of SDN + MANET. However, the main weakness of this work is its lack of consideration of mobility management commonly seen in MANET. Our work has similar technical approach for network protocol stacks bridging. In addition, our work takes one step further in that the SDN overlay not only offers data forwarding over the distributed underlay as does reference [46], but also provides practical centralized sensing and mobility control through extended OpenFlow, which is very valuable for MANET applications. Reference [47] proposed to bridge SDN domains with west-east interfaces in a peer-to-peer manner, which offered new possibilities for interconnectivity and collaboration between software-defined MANETs (i.e., SDN + MANET). However, unlike our work, this work mainly focuses on inter-domain SDN data forwarding, hence limited value in MANET applications. On the other hand, it does offer west-east interoperability between SDN domains, which can be combined with our work in the future for inter-MANET-swarm collaboration. In this paper, we primarily focus on the collaboration inside a MANET swarm.

3) IOT-BASED MONITORING AND MANAGEMENT

Our work can be reasonably extended to bridge SDN and IoT-related protocols such as ZigBee to construct an SDN-controlled IoT swarm which we are currently working on. Therefore, we summarize IoT-related works in the following two subsections. IoT is extensively used in monitoring and management in real world projects. Reference [48] proposed a safety and health monitoring system for outdoor construction workers. The proposed system consists of a wearable body area network (WBAN) to collect and transmit subjects' physiological metrics and ambient environmental statistics using BLE (Bluetooth Lower Energy), and a low-power wide area network (LPWAN) to connect the WBAN with the Internet through LoRa. Compared with our work, it does not have to handle mobility issues since the sensors are statically deployed on the subject body hence the fixed local WBAN topology. In addition, our work defers from this work in that our proposed solution applies in mobile ad hoc networking (i.e., dynamic topologies) without fixed infrastructures. IoT-based systems are also applied in hydrology/underwater-related monitoring and management. Swarm robotics are recently reported to be applied in underwater exploration [49]. Reference [50] surveyed the architecture, communication, applications and challenges in the field of underwater swarm robotics. Reference [51] designed a swarm of autonomous miniature underwater robot drifters for exploring submesoscale ocean dynamics. The swarm consists of 16 independent vehicles whose 3D trajectories are measured near-continuously underwater. These drifting vehicles can also perform vertical swimming behaviors to mimic plankton for biological study. Besides, vehicles can autonomously invoke depth-control algorithm for continuous subsurface measurements. The overall design goal was to create a portable, easily deployed system that could provide 3D localizations at reasonable update rates in a relevant volume of water. Unlike our work, it involves very little in-swarm communication. The swarm of independent vehicles constitutes a measurement array instead of a collaborative group. Reference [22] reported an IoT-based precision irrigation project in agriculture, named SWAMP (smart water management platform). SWAMP intends to improve the use of water resources by means of IoT in heterogeneous pilots (two in Europe and two in Brazil). This reference also summarized the security threats and risks that are faced with such a precision irrigation system, and suggested the application of SDN inside the system to deal with these threats. Our work differs from this work in that the software-defined

control over and collaboration between mobile nodes is the primary focus. Reference [52] proposed a decentralized cloud M2M (machine-to-machine) system for big data processing in renewable energy plants such as hydropower plants. Parameters are recorded by the fixed-positioned remote telemetry units (RTUs), and transmitted for sophisticated data processing. While this reference focuses on statistics collecting and data processing, our work focuses on the mobile nodes collaboration during mobile monitoring.

4) SDN INTEGRATION INTO IOT

To cope with IoT access problem, UbiFlow [23] proposed a unified flow control and access mobility management solution using SDN methodologies. Distributed SDN controllers are deployed for large-scale flow scheduling, fault-tolerance, etc., in IoT access network. UbiFlow is envisioned to be deployed in scenarios with fixed IoT infrastructures and mobile IoT clients. Our work defer from UbiFlow in that the proposed solution applies in mobile ad hoc networking without fixed infrastructures. Reference [24] proposed a data-centric IoT architecture in order to facilitate IoT data aggregation at data centers using SDN/NFV techniques, for IoT service orchestration, IoT QoS provisioning, etc. In this architecture, IoT resides on the edge network whereas SDN resides on the cloud, i.e., an architecture with two comparatively independent tiers. Our work defers from this work in that the proposed scheme focused on the deep integration of SDN into distributed underlay, especially on how a swarm can be efficiently controlled by means of SDN techniques in mobile and volatile topologies. Reference [21] envisioned the integration of cloud computing and IoT (i.e., CloudIoT) and proposed the SDN-powered 3-layered security evaluation framework for CloudIoT. On the basis of the integration of SDN and IoT, references [25] proposed to deploy deep learning agents in the centralized SDN controller to enhance system applicability in various scenarios such as smart city, industrial IoT, etc. These references focused on functionalities enrichment (e.g., security evaluation, machine learning) inside a large-scale IoT by means of SDN technologies, instead of the ad hoc networking and software-defined control of the swarm. Reference [53] coined the concept of Software-Defined Device, targeting at providing a unified software framework interfaces that encapsulate hardware-specifics as easily-invokable ''software'' components for rapid IoT application development and deployment. Our work defers from this work in that OpenFlow is used as the control method over IoT nodes, thus the enhancement of interoperability and compatibility with wired SDN networks.

VI. CONCLUSIONS

The distributed MANET underlay offers flexible moving capabilities and multifunctional sensing and monitoring capabilities. In this paper, an SDN overlay is built on top of the MANET underlay by means of SDN controllers and switches, so that consistent and coherent control, not only in data forwarding but also in physical actions of mobile nodes, can be deployed over the MANET underlay in a centralized manner. This approach simplifies the global control over a MANET swarm in a way that only the MANET swarm head that activates the SDN controller is involved for the manipulation of the whole swarm. The experiment results show that the SDN-controlled MANET swarm approach is practical, especially for mobile and collaborative monitoring such as hydropower plant live surveillance. Also the runtime performance is applicable. It offers an appealing solution for mobile and collaborative surveillance tasks.

In our future work, we are going to extend our SDN-controlled MANET swarm in the following ways:

- To extend the physical actions. Currently, the physical actions issued by OpenFlow are yet to be enlarged for complex scenarios such as harmful animal tracking around hydropower plants, etc. More actions will be derived and tested in our future works.
- To extend the vehicle platform. This work proved the feasibility and applicability of the SDN-controlled MANET swarm based on the wheeled Raspberry Pi hardware platform. Other vehicle platforms such as UAV will be considered in our future work.
- To control IoT underlay. Currently, MANET constitutes the distributed underlay by bridging the SDN and ad hoc protocols. We are now working on the bridging between SDN and ZigBee for short-range low-power wireless communications to construct the SDN-controlled IoT swarm for broader means of monitoring.
- To study the possibility of integrating machine learning techniques into the SDN-controlled MANET swarm node to improve the intelligence of surveillance capabilities that can smartly and automatically detect targets, leading to a more intelligent MANET swarm platform.

REFERENCES

[1] D. S. M. Corson and J. P. Macker, *Mobile Ad hoc Networking (MANET): Routing Protocol Performance Issues and Evaluation Considerations*. document RFC 2501, 1999. [Online]. Available: https://rfc-editor.org/rfc/rfc2501.txt

[2] D. Kreutz, F. Ramos, P. E. Veríssimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, ''Software-defined networking: A comprehensive survey,'' *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[3] W. Miao, F. Agraz, S. Peng, S. Spadaro, G. Bernini, J. Perelló, G. Zervas, R. Nejabati, N. Ciulli, D. Simeonidou, H. Dorren, and N. Calabretta, ''SDN-enabled OPS with QoS guarantee for reconfigurable virtual data center networks,'' *IEEE/OSA J. Opt. Commun. Netw.*, vol. 7, no. 7, pp. 634–643, Jul. 2015.

[4] G. Sun, D. Liao, D. Zhao, Z. Xu, and H. Yu, ''Live migration for multiple correlated virtual machines in cloud-based data centers,'' *IEEE Trans. Services Comput.*, vol. 11, no. 2, pp. 279–291, Apr. 2018.

[5] G. Sun, D. Liao, D. Zhao, Z. Sun, and V. Chang, ''Towards provisioning hybrid virtual networks in federated cloud data centers,'' *Future Gener. Comput. Syst.*, vol. 87, pp. 457–469, Oct. 2018.

[6] M. Soliman, B. Nandy, I. Lambadaris, and P. Ashwood-Smith, ''Exploring source routed forwarding in SDN-based WANs,'' in *Proc. IEEE Int. Conf. Commun. (ICC)*, Jun. 2014, pp. 3070–3075.

[7] G. Sun, V. Chang, G. Yang, and D. Liao, ''The cost-efficient deployment of replica servers in virtual content distribution networks for data fusion,'' *Inf. Sci.*, vol. 432, pp. 495–515, Mar. 2018.

[8] H. Yu, T. Wen, H. Di, V. Anand, and L. Li, ''Cost efficient virtual network mapping across multiple domains with joint intra-domain and inter-domain mapping,'' *Opt. Switching Netw.*, vol. 14, pp. 233–240, 2014. doi: 10.1016/j.osn.2014.05.020.

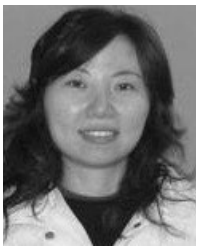
[9] H. Yu, C. Qiao, J. Wang, L. Li, V. Anand, and B. Wu, "Regional failure-resilient virtual infrastructure mapping in a federated computing and networking system," *IEEE/OSA J. Opt. Commun. Netw.*, vol. 6, no. 11, pp. 997–1007, Nov. 2014.
[10] X. Chen and T. Wu, "Towards the semantic Web based northbound interface for SDN resource management," in *Proc. IEEE 11th Int. Conf. Semantic Comput. (ICSC)*, San Diego, CA, USA, Jan./Feb. 2017, pp. 40–47. doi: 10.1109/ICSC.2017.27.
[11] G. Sun, F. Zhang, D. Liao, H. Yu, X. Du, and M. Guizani, "Optimal energy trading for plug-in hybrid electric vehicles based on fog computing," *IEEE Internet Things J.*, vol. 6, no. 20, pp. 2309–2324, Apr. 2019.
[12] X. Chen, J. Wu, and T. Wu, "The top-K QoS-aware paths discovery for source routing in SDN," *KSII Trans. Internet Inf. Syst.*, vol. 12, no. 6, pp. 2534–2553, 2018.
[13] X. Chen, T. Wu, and L. Xie, "The declarative and reusable path composition for semantic Web-driven SDN," *IEICE Trans. Commun.*, vol. E101.B, no. 3, pp. 816–824, Mar. 2018. doi: 10.1587/transcom.2017EBP3233.
[14] X. Chen, Z. Li, Y. Zhang, R. Long, H. Yu, X. Du, and M. Guizani, "Reinforcement learning–based QoS/QoE-aware service function chaining in software-driven 5G slices," *Trans. Emerg. Telecommun. Technol.*, vol. 29, no. 11, p. e3477, 2018. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/ett.3477
[15] K. Yang, H. Zhang, and P. Hong, "Energy-aware service function placement for service function chaining in data centers," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2016, pp. 1–6.
[16] T. Wen, H. Yu, and X. Du, "Performance guarantee aware orchestration for service function chains with elastic demands," in *Proc. IEEE Conf. Netw. Function Virtualization Softw. Defined Netw. (NFV-SDN)*, Nov. 2017, pp. 1–4.
[17] G. Sun, Y. Li, D. Liao, and V. Chang, "Service function chain orchestration across multiple domains: A full mesh aggregation approach," *IEEE Trans. Netw. Service Manag.*, vol. 15, no. 3, pp. 1175–1191, Sep. 2018.
[18] G. Sun, Y. Li, H. Yu, A. V. Vasilakos, X. Du, and M. Guizani, "Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks," *Future Gener. Comput. Syst.*, vol. 91, pp. 347–360, Feb. 2019. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167739X1831848X
[19] *IEEE Standard for Local and Metropolitan Area Networks—Station and Media Access Control Connectivity Discovery*, IEEE Standard802.1AB-2016 (Revision IEEE Std 802.1AB-2009), Mar. 2016, pp. 1–146.
[20] N. Mckeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling innovation in campus networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, Apr. 2008.
[21] Z. Han, X. Li, K. Huang, and Z. Feng, "A software defined network-based security assessment framework for CloudIoT," *IEEE Internet Things J.*, vol. 5, no. 3, pp. 1424–1434, Jun. 2018.
[22] C. Kamienski, J. Kleinschmidt, J.-P. Soininen, K. Kolehmainen, L. Roffia, M. Visoli, R. F. Maia, and S. Fernandes, "SWAMP: Smart water management platform overview and security challenges," in *Proc. 48th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. Workshops (DSN-W)*, Jun. 2018, pp. 49–50.
[23] D. Wu, D. I. Arkhipov, E. Asmare, Z. Qin, and J. A. McCann, "UbiFlow: Mobility management in urban-scale software defined IoT," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2015, pp. 208–216.
[24] A. Hakiri, P. Berthou, A. Gokhale, and S. Abdellatif, "Publish/subscribe-enabled software defined networking for efficient and scalable IoT communications," *IEEE Commun. Mag.*, vol. 53, no. 9, pp. 48–54, Sep. 2015.
[25] M. Mohammadi, A. Al-Fuqaha, M. Guizani, and J. Oh, "Semisupervised deep reinforcement learning in support of IoT and smart city services," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 624–635, Apr. 2018.
[26] M. Mahalingam, D. Dutt, K. Duda, P. Agarwal, L. Kreeger, T. Sridhar, M. Bursell, and C. Wright, *Virtual Extensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks*, document RFC 7348, 2014. [Online]. Available: https://www.rfc-editor.org/info/rfc7348
[27] P. Baronti, P. Pillai, V. W. C. Chook, S. Chessa, A. Gotta, and Y. F. Hu, "Wireless sensor networks: A survey on the state of the art and the 802.15.4 and ZigBee standards," *Comput. Commun.*, vol. 30, pp. 1655–1695, May 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0140366406004749
[28] M. Centenaro, L. Vangelista, A. Zanella, and M. Zorzi, "Long-range communications in unlicensed bands: The rising stars in the IoT and smart city scenarios," *IEEE Wireless Commun.*, vol. 23, no. 5, pp. 60–67, Oct. 2016.
[29] Y.-E. Wang, X. Lin, A. Adhikary, A. Grovlen, Y. Sui, Y. Blankenship, J. Bergman, and H. S. Razaghi, "A primer on 3GPP narrowband Internet of Things," *IEEE Commun. Mag.*, vol. 55, no. 3, pp. 117–123, Mar. 2017.
[30] S. R. Das, C. E. Perkins, and E. M. Belding-Royer, *Ad Hoc On-Demand Distance Vector (AODV) Routing*, document RFC 3561, Jul. 2003. [Online]. Available: https://rfc-editor.org/rfc/rfc3561.txt
[31] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in *Proc. ACM Conf. Commun. Archit., Protocols Appl. (SIGCOMM)*, 1994, pp. 234–244. doi: 10.1145/190809.190336.
[32] D. A. Maltz and D. C. Johnson, *The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4*, document IETF RFC 4728, Feb. 2007. [Online]. Available: https://rfc-editor.org/rfc/rfc4728.txt
[33] T. H. Clausen and P. Jacquet, *Optimized Link State Routing Protocol (OLSR)*, document RFC 3626, Oct. 2003. [Online]. Available: https://rfc-editor.org/rfc/rfc3626.txt
[34] B. Pfaff, J. Pettit, T. Koponen, E. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, and P. Shelar, "The design and implementation of open vSwitch," in *Proc. 12th USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, vol. 40. Oakland, CA, USA: USENIX Association, 2015, pp. 117–130.
[35] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, T. Koide, B. Lantz, B. O'Connor, P. Radoslavov, W. Snow, and G. Parulkar, "ONOS: Towards an open, distributed SDN OS," in *Proc. ACM 3rd Workshop Hot Topics Softw. Defined Netw. (HotSDN)*, New York, NY, USA, 2014, pp. 1–6. doi: 10.1145/2620728.2620744.
[36] J. Medved, R. Varga, A. Tkacik, and K. Gray, "OpenDaylight: Towards a model-driven SDN controller architecture," in *Proc. IEEE Int. Symp. World Wireless, Mobile Multimedia Netw. (WoWMoM)*, Jun. 2014, pp. 1–6.
[37] S. Panchen, N. McKee, and P. Phaal, *InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks*, document RFC 3176, 2001. [Online]. Available: https://rfc-editor.org/rfc/rfc3176.txt
[38] X. Chen, Y. Chen, A. K. Sangaiah, S. Luo, and H. Yu, "MonLink: Piggyback status monitoring over LLDP in software-defined energy Internet," *Energies*, vol. 12, no. 6, p. 1147, Jan. 2019. [Online]. Available: https://www.mdpi.com/1996-1073/12/6/1147
[39] K.-K. Yap, M. Kobayashi, R. Sherwood, T.-Y. Huang, M. Chan, N. Handigol, and N. McKeown, "OpenRoads: Empowering research in mobile networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 125–126, Jan. 2010. doi: 10.1145/1672308.1672331.
[40] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "NOX: Towards an operating system for networks," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 3, pp. 105–110, Jul. 2008. doi: 10.1145/1384609.1384625.
[41] P. Dely, A. Kassler, and N. Bayer, "OpenFlow for wireless mesh networks," in *Proc. 20th Int. Conf. Comput. Commun. Netw. (ICCCN)*, Jul./Aug. 2011, pp. 1–6.
[42] K.-K. Yap, R. Sherwood, M. Kobayashi, T.-Y. Huang, M. Chan, N. Handigol, N. McKeown, and G. Parulkar, "Blueprint for introducing innovation into wireless mobile networks," in *Proc. 2nd ACM SIGCOMM Workshop Virtualized Infrastruct. Syst. Archit. (VISA)*, New York, NY, USA, 2010, pp. 25–32. doi: 10.1145/1851399.1851404.
[43] A. Detti, C. Pisa, S. Salsano, and N. Blefari-Melazzi, "Wireless Mesh Software Defined Networks (wmSDN)," in *Proc. IEEE 9th Int. Conf. Wireless Mobile Comput., Netw. Commun. (WiMob)*, Oct. 2013, pp. 89–95.
[44] P. Baskett, Y. Shang, W. Zeng, and B. Guttersohn, "SDNAN: Software-defined networking in ad hoc networks of smartphones," in *Proc. IEEE Consum. Commun. Netw. Conf. (CCNC)*, Jan. 2013, pp. 861–862.
[45] I. Ku, Y. Lu, M. Gerla, F. Ongaro, R. L. Gomes, and E. Cerqueira, "Towards software-defined VANET: Architecture and services," in *Proc. 13th Annu. Medit. Ad Hoc Netw. Workshop (MED-HOC-NET)*, Jun. 2014, pp. 103–110.
[46] H. C. Yu, G. Quer, and R. R. Rao, "Wireless SDN mobile ad hoc network: From theory to practice," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–7.
[47] P. Lin, J. Bi, Z. Chen, Y. Wang, H. Hu, and A. Xu, "WE-bridge: West-east bridge for SDN inter-domain network peering," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr./May 2014, pp. 111–112.

[48] F. Wu, T. Wu, and M. R. Yuce, "An Internet-of-Things (IoT) network system for connected safety and health monitoring applications," *Sensors*, vol. 19, no. 1, p. 21, Jan. 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/1/21
[49] H. Luo, K. Wu, R. Ruby, Y. Liang, Z. Guo, and L. M. Ni, "Software-defined architectures and technologies for underwater wireless sensor networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2855–2888, 4th Quart., 2018.
[50] K. K. S, B. Mahapatra, and V. G. Menon, "Into the world of underwater swarm robotics: Architecture, communication, applications and challenges," *Recent Patents Comput. Sci.*, vol. 12, no. 1, pp. 25–37, 2019.
[51] J. S. Jaffe, P. J. S. Franks, P. L. D. Roberts, D. Mirza, C. Schurgers, R. Kastner, and A. Boch, "A swarm of autonomous miniature underwater robot drifters for exploring submesoscale ocean dynamics," *Nature Commun.*, vol. 8, Jan. 2017, Art. no. 14189. [Online]. Available: https://www.nature.com/articles/ncomms14189
[52] G. Suciu, A. Vulpe, A. Martian, S. Halunga, and D. N. Vizireanu, "Big data processing for renewable energy telemetry using a decentralized cloud M2M system," *Wireless Pers. Commun.*, vol. 87, no. 3, pp. 1113–1128, Apr. 2016. doi: 10.1007/s11277-015-2527-7.
[53] P. Hu, H. Ning, L. Chen, and M. Daneshmand, "An open Internet of Things system architecture based on software-defined device," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2583–2592, Apr. 2019.

XI CHEN received the B.S. and Ph.D. degrees in computer science from Southwest Jiaotong University, China, in 2007 and 2013, respectively. From 2011 to 2012, he studied as a Visiting and joint Ph.D. candidate from the School of Computer Science and Engineering, University of New South Wales (UNSW), Australia. He is currently with the School of Computer Science and Technology, Southwest Minzu University, China. He also holds a postdoctoral position at the School of Information and Communication Engineering, University of Electronic Science and Technology of China. His research interests include software-defined networking (SDN), network function virtualization (NFV), service function chaining (SFC), cloud computing, computer networks, and web services.

TAO WU received the B.S. and Ph.D. degrees in computer science from Southwest Jiaotong University, China, in 2007 and 2014, respectively. She is currently a Lecturer with the School of Computer Science, Chengdu University of Information Technology, China. Her research interests include particle swarm optimization (PSO) and wireless networks.

GANG SUN was a Visiting Fellow with The Australian National University, from 2015 to 2016. He is currently an Associate Professor of computer science with the University of Electronic Science and Technology of China (UESTC). He has coauthored 80 technical publications, including article in refereed journals and conferences, invited articles and presentations, and book chapters. His research interests include network virtualization, cloud computing, high performance computing, parallel and distributed systems, ubiquitous/pervasive computing and intelligence, and cyber security. He is also a member of the IEEE Computer Society. He has served as a Reviewer for the IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, the IEEE COMMUNICATIONS LETTERS, *Information Fusion*, *Future Generation Computer Systems*, the *Journal of Network and Computer Applications*, the *Journal of Supercomputing*, the *Journal of Parallel and Distributed Computing*, *KSII Transactions on Internet and Information Systems*, *Computers and Electrical Engineering*, and the *Chinese Journal of Electronics*. He has also edited special issues at top journals, such as *Future Generation Computer Systems* and *Multimedia Tools and Applications*.



HONGFANG YU received the B.S. degree in electrical engineering from Xidian University, in 1996, and the M.S. and Ph.D. degrees in communication and information engineering from the University of Electronic Science and Technology of China, in 1999 and 2006, respectively. From 2009 to 2010, she was a Visiting Scholar with the Department of Computer Science and Engineering, University at Buffalo (SUNY). She is currently a Professor and a Ph.D. Supervisor with the School of Information and Communication Engineering, University of Electronic Science and Technology of China. Her research interests include software-defined networking (SDN), network function virtualization (NFV), service function chaining (SFC), and cloud computing.