energies

MDPI

Article

MonLink: Piggyback Status Monitoring over LLDP in Software-Defined Energy Internet

Xi Chen [1,2], Yue Chen [3], Arun Kumar Sangaiah [4], Shouxi Luo [5] and Hongfang Yu [2,*,†]

1 School of Computer Science and Technology, Southwest Minzu University, Chengdu 610041, China; cx@swun.edu.cn
2 School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
3 National Computer Network Emergency Response Technical Team/Coordination Center of China, Beijing 100029, China; chenyue@cert.org.cn
4 School of Computing Science and Engineering, Vellore Institute of Technology, Tamil Nadu 632014, India; sarunkumar@vit.ac.in
5 School of Information Science and Technology, Southwest Jiaotong University, Chengdu 611756, China; sxluo@swjtu.edu.cn
* Correspondence: yuhf@uestc.edu.cn
† Current address: School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China.

Received: 6 January 2019; Accepted: 20 March 2019; Published: 25 March 2019

Abstract: While software-defined networking (SDN) has been widely applied in various networking domains including datacenters, WANs (Wide Area Networks), QoS (Quality of Service) provisioning, service function chaining, etc., it also has foreseeable applications in energy internet (EI), which envisions an intelligent energy industry on the basis of (information) internet. Global awareness provided by SDN is especially useful in system monitoring in EI to achieve optimal energy transportation, sharing, etc. Link layer discovery protocol (LLDP) plays a key role in global topology discovery in software-defined energy internet when SDN is applied. Nevertheless, EI-related status information (power loads, etc.) is not collected during the LLDP-based topology discovery process initiated by the SDN controller, which makes the optimal decision making (e.g., efficient energy transportation and sharing) difficult. This paper proposes MonLink, a piggyback status-monitoring scheme over LLDP in software-defined energy internet with SDN-equipped control plane and data plane. MonLink extends the original LLDP by introducing metric type/length/value (TLV) fields so as to collect status information and conduct status monitoring in a piggyback fashion over LLDP during topology discovery simultaneously without the introduction of any newly designed dedicated status monitoring protocol. Several operation modes are derived for MonLink, namely, periodic MonLink, which operates based on periodic timeouts, proactive MonLink, which operates based on explicit API invocations, and adaptive MonLink, which operates sensitively and self-adaptively to status changes. Various northbound APIs are also designed so that upper layer network applications can make full use of the status monitoring facility provided by MonLink. Experiment results indicate that MonLink is a lightweight protocol capable of efficient monitoring of topological and status information with very low traffic overhead, compared with other network monitoring schemes such as sFlow.

Keywords: software-defined networking (SDN); energy internet (EI); status monitoring; link layer discovery protocol (LLDP)

1. Introduction

J. Rifkin first put forward the concept of energy internet (EI) in his work about the third industrial revolution that he believed will happen in the foreseeable future [1]. Five pillars must be constructed before the revolution takes place: (1) renewable energy; (2) micro power plants converted from buildings for energy recycling; (3) hydrogen-based energy storage; (4) energy sharing based on internet technologies; (5) new energy vehicles, which comprehensively pave the foundation for the third industrial revolution.

Energy internet will comprise the backbone and infrastructure for the intelligent energy generation, harvesting, storage, management, transportation, sharing, etc. Similarities are shared between (information) internet and energy internet. While internet has routers to steer traffic between different local area networks (LANs), EI has power routers to steer energy between different power distribution networks (PDNs). Similarly, while the internet has switches to steer traffic inside a LAN, EI has power switches to steer energy inside a PDN. Therefore, facilitating EI on the basis of Internet is a natural choice. However, energy internet has far more complex content than that of internet. While the internet plays a role as the information hub with high-density computational, storage, and network resources, energy internet integrates energy industry, IoT, sensor technologies, big data, low-power WAN, etc., on the basis of internet. Thus simply expanding the internet to the energy field is far from addressing issues faced with EI. Therefore, various EI-related projects have been proposed by different countries such as FREEDM (Future Renewable Electric Energy Delivery and Management) in US [2,3], e-energy in Germany [4], etc.

Energy internet will also extend the information frontier and deepen the information depth beyond the reach of traditional internet. For example, for those remotely located renewable energy plants (such as wind power and photovoltaic power plants) sensitive to weather conditions, system status and electricity outcome are time-variant thus requiring long-term monitoring. Meanwhile, sensors connected to photovoltaic or wind power generators have limited battery thus requiring low-power transmission and well-tuned sampling intervals for long-term surveillance. In addition, for remotely located renewable energy plants, human configuration at site might not be possible due to severe natural conditions. It is a desirable feature if smart power equipments can be discovered (together with their system status information), monitored and controlled by a smart central hub once they are installed possibly through some automated mechanics. To summarize, automatic equipment discovery, low-power adaptive sensing, continuous real-time monitoring of the entire power plant and smart decision making will play key roles for the sustainable operation of energy internet. This requires new capabilities, especially control and management capabilities, to be added to the current internet to facilitate energy internet.

Despite the differences, EI and internet share similarities on structure backbone and networking mechanisms. New advancements in internet can be reasonably borrowed and adapted to EI. Software-defined networking (SDN) [5] is widely adopted in communication networks such as WAN (Wide Area Network) [6,7], datacenters [8–11], NFV (Network Function Virtualization) [12], network virtualization [13,14], service provisioning [15–20]. SDN introduces logically centralized software controller with elaborate control functions to enable smarter traffic steering, policy enforcement, global management over the whole network, etc. Therefore, methodologies in SDN can be applied in EI to control intelligent sensors, power routers/switches, thus a software-defined energy internet with SDN-equipped control plane and data plane that is capable of intelligent energy manipulation.

SDN was originally proposed by Stanford University to support the design of a network innovation platform based on the OpenFlow protocol [21,22], an open protocol that aims to break down vendor-specific barriers to facilitate network innovation. Among others, OpenFlow is the most accepted southbound interface (i.e., the controller-switch interface) for the controller to manage underlay switches that are OpenFlow-compatible. OpenFlow reveals the programmability of network devices, especially switches, by providing standardized programming interfaces. In addiction to the powerful programmability and flexible architecture, SDN is also equipped with the global topological view based on which globally optimized network resource utilization and decision making become possible. Link layer discovery is the foundation for the retrieval of global topological view and related information. The IEEE-proposed link layer discovery protocol (LLDP) [23] is the primary link layer discovery protocol used by SDN controllers. Since SDN works in a centralized fashion, LLDP-collected topological information is aggregated into the centered controller.

Should time-variant equipment metrics be piggybacked by LLDPDUs (Link Layer Discovery Protocol Data Units) traveling around a software-defined energy internet, system status information can be possibly aggregated into the controller during topology discovery to support lightweight and continuous monitoring in an energy internet, without drastic modification to the current system structure. Based on this thought, this paper proposes MonLink, the piggyback monitoring over LLDP for software-defined energy internet, so that the lightweight status-aware topology discovery becomes possible. MonLink will work as the status information monitoring infrastructure that supports various status-aware applications. The contribution of this paper includes as follows:

- MonLink monitors the system status information in a piggyback fashion over LLDP during topology discovery. Since LLDP is widely adopted for topology discovery in various controllers, the usage of LLDP as the fundamental retrieval method for status information is "transparent" to SDN networks including software-defined energy internet.
- MonLink takes advantage of the extensibility of LLDP. It does not introduce any newly designed dedicated status monitoring protocol, thus very little traffic overhead during status monitoring. Therefore, it does not affect or compromise the working process of southbound interface of SDN (e.g., OpenFlow), i.e., a non-invasive design. (see Sections 2.2 and 2.7)
- MonLink works in a hybrid fashion that comprehensively considers self-adaptiveness and high interactiveness, meeting the requirements of time-variant and continuous monitoring in energy internet. (see Sections 2.3–2.6)
- MonLink makes use of the centralized control in SDN. No extra network monitoring system (such as sFlow [24]) is required to implement status information collecting (see Section 3.2).

The rest of this paper is organized as follows: Section 2 explains the protocol extension, design, and workflow of MonLink. It also specifies several working modes of MonLink that fit in different application scenarios to achieve flexible status information collecting. Section 3 compares MonLink with sFlow-based status information collecting through extensive experiments. Related works are summarized in Section 4. Finally, this paper is concluded in Section 5.

2. Materials and Methods

2.1. Software-Defined Energy Internet

In this paper, software-defined energy internet refers to the the SDN-equipped energy internet. Specifically, it is an energy internet equipped with the SDN control plane (i.e., OpenFlow-compatible SDN controllers) and SDN data plane (forwarding devices that support OpenFlow protocol). SDN controllers are easy to be deployed in energy internet since they are independent software entities. On the other hand, the deployment of SDN data plane involves two aspects: (1) the deployment of OpenFlow (power) switches/routers inside a renewable or traditional power plant; and optionally, (2) the enhancement of sensors (temperature sensors, humidity sensors, cameras, etc.) and device controllers (e.g., battery controllers) with OpenFlow agents (e.g., the software-implemented open vSwitch (OVS) [25]) installed inside these sensory terminals. If aspect (2) is additionally implemented, sensors and controllers are also part of the SDN data plane thus can be controlled directly by SDN controllers. This can enable an end-to-end (i.e., from the central SDN controller to edge power generators) controllable smart energy internet, simply using OpenFlow PDUs (Protocol Data Units) such as flow table entries with directive messages (i.e., flow rules) that sensors and controllers should obey. If only aspect (1) is implemented, status information of power generators collected by sensors can still be aggregated by OpenFlow (power) switches, and then sent to SDN controllers that are capable of making smart decisions based on the collected status information, and controlling edge power generators through other dedicated protocols. Either way enhances the controllability and capabilities of smart decision-making in energy internet, although the SDN-controlled coverage is different.

Figure 1 demonstrates the composition of the software-defined energy internet. The right part is a photovoltaic power plant network, i.e., a software-defined energy LAN. Sensors and device controllers are indirectly controlled by the SDN controller by means of OpenFlow protocol through wired (e.g., Ethernet) or wireless (LoRa, NB-IoT, ZigBee, etc.) channels, possibly protected by secure methods [26,27] in case of the need for security. Many such software-defined energy LANs are connected by OpenFlow power routers to constitute the software-defined energy internet (the left part). Applications can be built on top of energy internet by invoking various northbound interfaces (green dashed lines) provided by SDN controllers. Therefore, the key to revealing the full potential of software-defined energy internet is a rich set of SDN-equipped control functions oriented to power devices, which are exposed by corresponding northbound interfaces. The remainder of this paper describes one such control function, MonLink, the piggyback status monitoring over LLDP in software-defined Energy Internet.

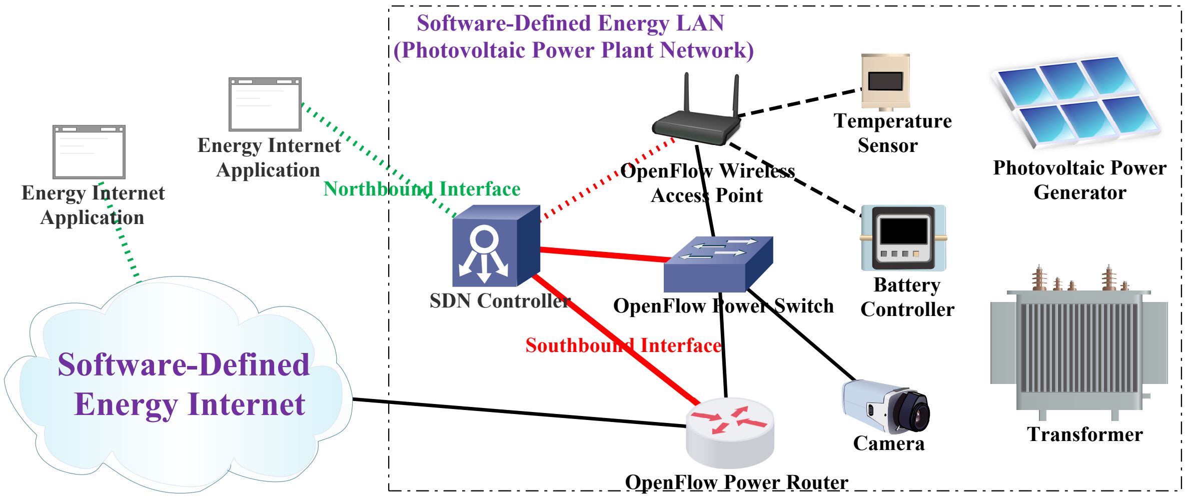

Software-Defined Energy LAN
(Photovoltaic Power Plant Network)
Energy Internet Application
Energy Internet Application
Northbound Interface
OpenFlow Wireless Access Point
Temperature Sensor
Photovoltaic Power Generator
SDN Controller
OpenFlow Power Switch
Battery Controller
Southbound Interface
Software-Defined Energy Internet
Camera
Transformer
OpenFlow Power Router

Figure 1. Software-defined energy internet.

2.2. Protocol Extension to LLDP

Although the SDN controller maintains the awareness of the existence of directly connected switches by sending and receiving "hello" messages, it is not aware of the topological structure formed by interconnected switches. To acquire the topological knowledge, the SDN controller must conduct the link layer discovery so as to reveal how underlying switches are mutually connected once the SDN network is initiated. The IEEE-proposed LLDP is the de facto protocol to fulfill this centralized topology discovery in SDN field.

LLDP works along with the flow matching mechanism in SDN. A packet_out message is sent by the controller to one of the connected switches to instruct it to multicast the LLDP packet contained in the packet_out. Inside the LLDP packet, basic information about this switch (such as MAC address) is included. All other switches (i.e., receiver switches) connected to this sender switch receive and match this LLDP packet against their own flow tables. At this point of time, no dedicated flow table entries are installed for LLDP traffic, so that this mis-matched LLDP packet is sent to the controller by means of packet_in to inquire how it should be processed. Since the packet_in contains both the sender and receiver switches' MAC (Media Access Control) addresses, etc., it is reasonable for the controller to assert that there exists a link between these two switches. This LLDP multicast process instructed by the controller is periodically carried out throughout the lifespan of the SDN network so that the global topology as well as its continuous updates are captured in a timely fashion. If extra status information is somehow included in LLDP packets in addiction to basic information such as MAC addresses, status monitoring can occur simultaneously with topology discovery/maintenance.

This is especially appealing for power device/equipment discovery and continuous monitoring when the SDN methodologies are deployed in the energy internet. The key lies in how status information can be incorporated. Thanks to the extensibility offered by LLDP, status information can be encapsulated inside TLV (Type/Length/Value, i.e., key-value pair with length information) fields. The string value subfield is where extra information resides, as shown in the "Metrics" in Figure 2. Currently, we consider communication-centric properties such as delay, bandwidth, packet loss, and jitter, 8 bytes for each metric. Note that metrics and length assigned for each metric can vary according to application configurations. For example, metrics such as temperature, humidity, battery, etc., can be included for energy internet oriented applications. We call this piggybacked status monitoring on the basis of LLDP MonLink.

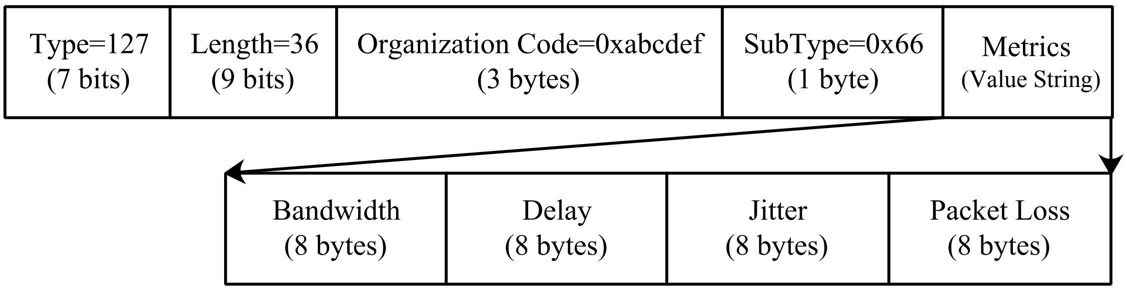


Figure 2. MonLink packet format.

In order to achieve flexible status information collection in different application scenarios, several working modes of MonLink were designed: periodic MonLink, which is triggered by timeouts, adaptive MonLink, which is an adaptive version of periodic MonLink, and proactive MonLink, which is triggered by explicit user invocation, to adapt our scheme to different network environments and requirements.

2.3. Periodic MonLink

The periodic MonLink follows the convention of the standard LLDP that LLDPDUs are sent out by the controller on a periodic basis during the life cycle of an SDN network. Once a switch receives an LLDPDU, periodic MonLink is carried out along with the topology discovery. The disadvantage of periodic MonLink is that LLDP multicast interval might not be short enough for scenarios where metrics fluctuate violently and frequently. During the LLDP multicast interval, status database in the controller is not updated. The worst case is that metrics fluctuation happens during the interval (i.e., between two multicasts) and recovers at almost the same time the next multicast happens (usually in a scale of several milliseconds). In this case, the collected status information is almost identical to that collected at the previous round, failing to capture the dynamics and violent metrics fluctuation between two multicasts.

The previous problem can be relieved using reduced LLDP multicast interval. However, smaller interval means more frequent sending of LLDP multicasts which in turn result in heavier traffic overhead caused by MonLink. This would consume larger bandwidth meant for user traffic since the underlay network is an in-band legacy network where overhead traffic including LLDP is transmitted along with user-data traffic in the same data plane. Thus the optimal interval relies on deep analysis of historical trending of status changes so that the LLDP multicast interval can be fine-tuned to better match the status collecting. Therefore, APIs dedicated for interval fine-tuning is provided by MonLink. However, to determine an appropriate multicast interval is still a challenging task, requiring both experience and deep theoretical study. In order to enhance adaptiveness and interactiveness, two more working modes of MonLink are derived.

2.4. Proactive MonLink

In order to cope with the disadvantage faced with periodic MonLink, we specifically designed the proactive MonLink. Individual monitoring requests might come to the SDN network in an ad hoc fashion, thus the SDN controller must be able to acquire the status information or update the previous status database on demand, especially for time-critical scenarios. Periodic MonLink cannot obtain status information on demand. Therefore, proactive MonLink is designed to fulfill the on-demand status information collecting. This working mode offers the chance for proactive interference by network administrators or applications. Therefore, APIs must be given on top of the SDN controller for the network administrators or applications to initiate a new round of status information collecting. Whenever an ad hoc status monitoring request comes, the controller can be automatically triggered to start a new round of MonLink multicast. In a word, proactive MonLink complements periodic MonLink with explicit interference from upper-layer entities (network applications, network administrators, etc.).

2.5. Adaptive MonLink

As mentioned above, to determine the appropriate sampling intervals between any two consecutive MonLink multicasts was non-trivial work. It was a desirable feature if the sampling intervals can be determined automatically and adaptively. To achieve this, MonLink must work in a stateful manner, keeping records of currently collected status information and deciding intervals accordingly by elaborating adaptive algorithms. What was expected was that intervals automatically shrank when status information varied violently in a short term while intervals steadily expand to initial values when status information remained almost unchanged. This required short-term prediction based on the current status. The core of the adaptive interval algorithm is shown in Equation (1).

$$I_i(t) = I_{base} \cdot e^{-\sqrt{\sum_{j=1}^{N}\left(W_j \cdot \frac{S_{i,j}(t)-S_{i,j}(t-1)}{S_{bench,j}}\right)^2}} + I_{min} \tag{1}$$

It is a Euclidean distance based decay function that predicts interval of the next MonLink iteration. Table 1 specifies the notations used in Equation (1) and Algorithm 1 (i.e., the adaptive MonLink). Inside the square root part is the Euclidean distance between the current status information and the status collected at the previous iteration.

Table 1. Notations and definitions for adaptive MonLink.

Notations	Definitions
M	The number of MonLink monitored equipments
N	The dimensionality of status information
$I_{periodic}$	The sampling interval for periodic MonLink
I_{min}	The minimum sampling interval for adaptive MonLink
I_{base}	The base sampling interval for adaptive MonLink
$I_i(t)$	The calculated interval for the i-th monitored equipment at iteration t
$IS(t)$	The set to store all calculated intervals for monitored equipments at iteration t
$I_{mc}(t)$	The multicast interval for adaptive MonLink
S_{bench}	The j-dimensional status benchmark
$S_{bench,j}$	The j-th dimension of S_{bench}
$S_i(t)$	The j-dimensional status information for the i-th monitored equipment at iteration t
$S_{i,j}(t)$	The j-th dimension of $S_i(t)$
W_j	The weight of the j-th dimension of status information

The reason why the Euclidean distance was used is due to its tendency irrelevance, meaning that, no matter if a metric increases or decreases, the change can be correctly captured using the same equation. Metrics can be favored using different weights (i.e., W_j) for distance computation. Meanwhile, as seen in Equation (1), we did not merely use Euclidean distance; instead, a Euclidean distance-based decay function was also used. The reason why the interval was finally determined by a decay function instead of just the Euclidean distance was that the non-linear decay function varied much faster than the linear Euclidean distance. If changes of status indicated by Euclidean distance were detected at present time, exhibiting the initial trend of system changes, MonLink must get ready for this trend in advance using even smaller intervals determined by the decay function. Therefore, short-term prediction became possible. Once the system status became stable again (i.e., almost unchanged), regardless of the absolute values, the interval automatically increased back to be around maximum (i.e., $I_{base} + I_{min}$) due to that $e^{-d} \to 1$ given $d \to 0$, where d is the Euclidean distance. This is how adaptivity becomes possible based on Equation (1).

Besides, the status value difference $\left(S_{i,j}(t) - S_{i,j}(t-1)\right)$ is divided by a status benchmark value $S_{bench,j}$ in Equation (1). Therefore, status benchmark settings significantly affected the sensibility of adaptive MonLink. Smaller settings led to an adaptive MonLink which was much more sensitive to minor status changes, due to the non-linearity of the exponential function e^{-d}. However, if the benchmark settings are too small (resulting in greater Euclidean distance thus smaller interval), two side-effects might occur: (1) too frequent MonLink multicasts might cause overwhelming LLDP traffic to compete the bandwidth; (2) the interval is so small that there might not be enough time for MonLink multicasts to take effect. Therefore, we elaborately provide the minimum interval I_{min} as the additive factor (the immutable part) in Equation (1) to ensure MonLink multicasts work properly to successfully collect status information. Presently, I_{min} is set to one second. The mutable part of the interval was determined by the decayed base interval I_{base} to capture the system variation, where $I_{base} = I_{periodic} - I_{min} = 14$. Currently, we manually configure the status benchmark settings according to experience. Nevertheless, it requires expertise in understanding and tuning status benchmark settings. Even an experienced network administrator can hardly give settings that fully capture the network dynamics. Therefore, we will further refine the status benchmark setting methods by means of machine learning techniques in future works, especially the hyper parameter tuning [28]. Specifically, we have the plan that makes use of the deep reinforcement learning (DRL) [29] to fit the values of benchmarks. DRL combines the reinforcement learning with the deep neural network (DNN) so that it learns the optimal policy by estimating the cumulative rewards gained from the current action committed to the environment. Should the settings of benchmarks be considered as the action and the accuracy of the monitoring as the reward, DRL is able to iteratively optimize the policy on how the values of benchmarks can be set in an automatic, adaptive and online manner. The core to design DRL for benchmarks evaluation lies in the modeling of reward that reflects the monitoring accuracy, on which we will conduct a deep research in the recent future work. Concretely, deep deterministic policy gradient (DDPG) [30] is considered a feasible candidate framework to generate continuous benchmarks value estimation. Table 2 exhibits the current interval and status benchmark settings.

Table 2. An example of adaptive MonLink settings.

	Properties	Notations	Values	Unit
Interval Settings	Periodic Interval	$I_{periodic}$	15	s
	Mininum Interval	I_{min}	1	s
	Base Interval	I_{base}	14	s
Status Benchmark	Bandwidth	$S_{bench,1}$	20,000,000	bps
	Delay	$S_{bench,2}$	100	µs
	Packet Loss	$S_{bench,3}$	0.05	N/A
	Jitter	$S_{bench,4}$	50	µs

Algorithm 1 gives the details of how adaptive MonLink works. For each iteration, status information of every monitored equipment is collected to compute the equipment-specific MonLink interval. The smallest of all equipment-specific MonLink intervals was selected as the MonLink multicast interval to conduct next round MonLink monitoring.

Algorithm 1 Adaptive MonLink.

```
1:  initialize $I_{min}$, $I_{base}$, $I_{periodic}$
2:  initialize $j$-dimensional status benchmark $S_{bench}$
3:  $I_{mc}(1) = I_{periodic}$
4:  for $(t = 1; ; t++)$ do
5:      wait for $I_{mc}(t)$ seconds before MonLink multicast
6:      $IS(t+1) = \emptyset$
7:      for $(i = 1; i \leq M; i++)$ do
8:          collect $j$-dim $S_i(t)$ for the $i$-th monitored equipment
9:          calculate next interval $I_i(t+1)$ based on Equation (1)
10:         $IS(t+1) = IS(t+1) \cup \{I_i(t+1)\}$
11:     end for
12:     $I_{mc}(t+1) = \min(IS(t+1))$
13: end for
```

To illustrate how adaptive MonLink works, an example is given in Table 3. Suppose a switch had varying metrics, e.g., bandwidth, delay, packet loss, jitter, etc., where every metric had equal weight $W_j = 1$ (weights can also be normalized). At iteration one, adaptive MonLink interval was the same as periodic MonLink interval (15 s), and the status information of the monitored switch is shown in the first line of Table 3. For violently changed metrics, greater decay caused by longer Euclidean distance resulted in much shorter intervals at the current iteration so as to capture system dynamics. For example, bandwidth (125 Mpbs) in iteration three was much greater that of iteration two (99 Mbps), thus a very small interval as short as 6 s. On the contrary, mildly changed metrics caused only minor interval changes (14 s) interval at iteration two compared with iteration one. Note also that at iteration six, collected status information was the same as that of iteration five, therefore, the interval increases back to the original value 15 s.

2.6. The Workflow

The overall workflow of MonLink that combines the periodic, adaptive and proactive modes are shown in Figure 3. When running MonLink, at least two threads must be launched, one for the invocation-triggered sequences (i.e., the proactive MonLink) and one for the time-triggered sequences (i.e., the periodic MonLink and adaptive MonLink), so that different modes can work together simultaneously to achieve greater flexibility. The gray sequences in Figure 3 are the core MonLink functions shared by periodic, adaptive and proactive modes, whereas blue ones, yellow ones and green ones are periodic, proactive and adaptive MonLink sequences, respectively.

2.7. The Implementation and APIs

The MonLink should be deployed at southbound interface where LLDP takes effect. That is to say both the controller and the switch must be modified to support MonLink. Specifically, the controller must be able to generate empty metric TLVs for the switches to fill in as well as to properly parse metric TLVs. Thus the code at the controller side governing LLDPDUs generation and parsing must be modified. Meanwhile, switches must be able to properly response to LLDPDUs containing metric TLVs. Therefore, corresponding code must be modified at switch side as well. Thanks to the idea that MonLink works in a piggyback fashion based on LLDP, the code modification was quite brief. We implemented MonLink using Floodlight [31] controller and Open vSwitch (OVS) [25].

In our implementation, about 20 lines of code were modified in OVS and about 60 lines of code modified in Floodlight to support basic functionalities of MonLink. Although code modification seems brief, to determine the right places for modification and re-compilation were challenging and subtle. Besides, various REST (Representational State Transfer) APIs were implemented for easier invocation of MonLink functionalities. Based on these REST APIs and backend code modification at both controller and switch sides, the MonLink platform was implemented as shown in Figure 4. Figure 4 exhibits the real-time status information (bandwidth, delay, jitter and packet loss) of a switch port. Also, upper-layer status-sensitive applications can be built on top of MonLink by invoking REST APIs ranging from basic status information collecting, optimal path planning, to more sophisticated traffic steering.

Table 3. An example of adaptive MonLink intervals.

Round	Metrics	Value	Distance	Decay	Interval (s)
1	Bandwidth Delay Packet Loss Jitter	100 Mbps 10 µs 0.005 1 µs	0	1	15
2	Bandwidth Delay Packet Loss Jitter	99 Mbps 9 µs 0.004 0.9 µs	0.055	0.947	14
3	Bandwidth Delay Packet Loss Jitter	120 Mbps 11 µs 0.005 2 µs	1.051	0.35	6
4	Bandwidth Delay Packet Loss Jitter	95 Mbps 10 µs 0.005 1 µs	1.25	0.286	5
5	Bandwidth Delay Packet Loss Jitter	100 Mbps 10 µs 0.005 5 µs	0.262	0.769	12
6	Bandwidth Delay Packet Loss Jitter	100 Mbps 10 µs 0.005 5 µs	0	1	15
7	Bandwidth Delay Packet Loss Jitter	99 Mbps 7 µs 0.07 2 µs	1.303	0.272	5
8	Bandwidth Delay Packet Loss Jitter	100 Mbps 10 µs 0.005 1 µs	1.301	0.272	5
9	Bandwidth Delay Packet Loss Jitter	100 Mbps 11 µs 0.005 1 µs	0.01	0.99	15

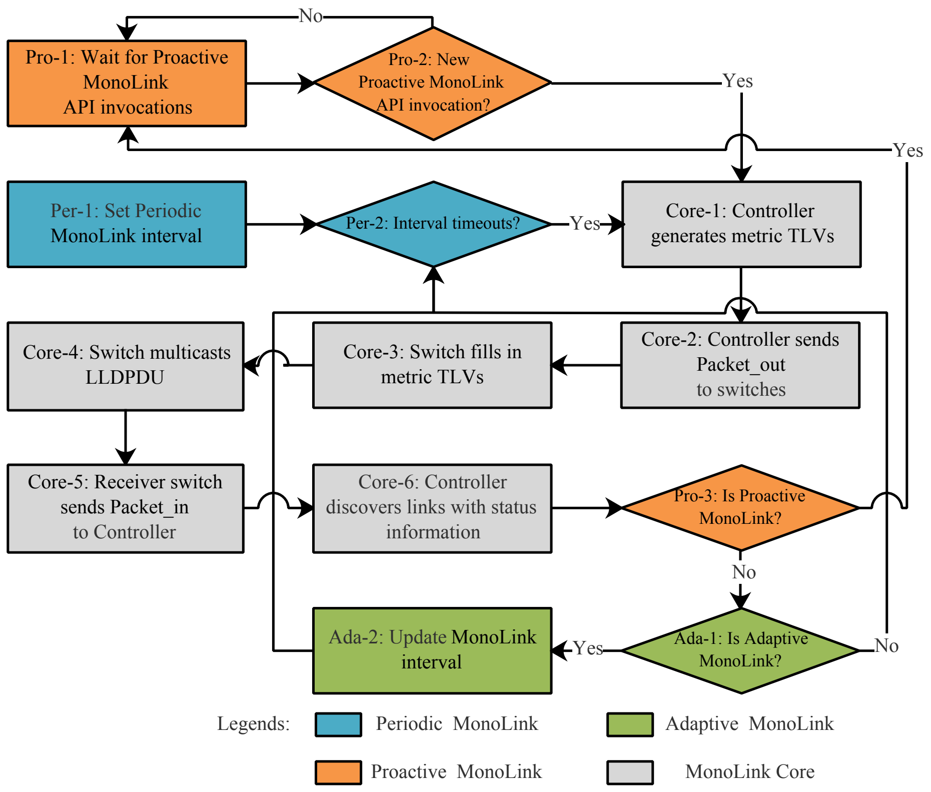

Pro-1: Wait for Proactive MonoLink API invocations
Pro-2: New Proactive MonoLink API invocation?
No
Yes
Yes
Per-1: Set Periodic MonoLink interval
Per-2: Interval timeouts?
Yes
Core-1: Controller generates metric TLVs
Core-2: Controller sends Packet_out to switches
Core-3: Switch fills in metric TLVs
Core-4: Switch multicasts LLDPDU
Core-5: Receiver switch sends Packet_in to Controller
Core-6: Controller discovers links with status information
Pro-3: Is Proactive MonoLink?
No
Ada-1: Is Adaptive MonoLink?
Yes
No
Ada-2: Update MonoLink interval
Legends:
Periodic MonoLink
Adaptive MonoLink
Proactive MonoLink
MonoLink Core

Figure 3. The workflow of MonLink.

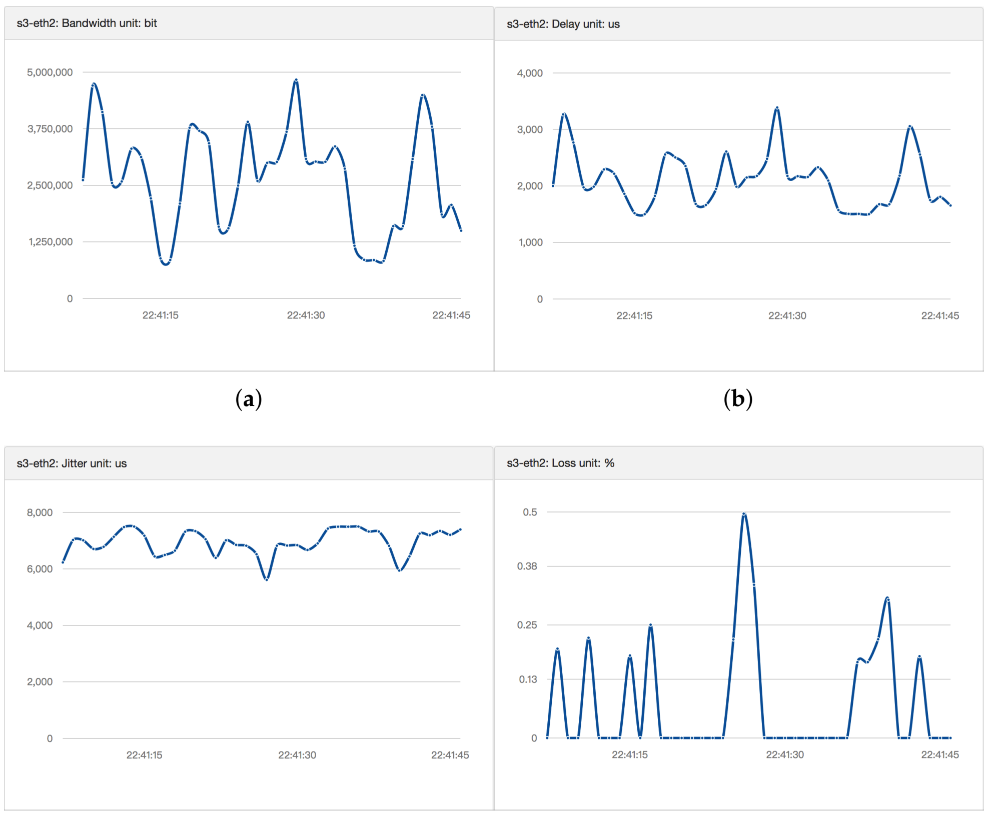


Figure 4. MonLink platform, the statistics page. (**a**) The bandwidth statistics. (**b**) The delay statistics. (**c**) The jitter statistics. (**d**) The packet loss statistics.

3. Results

Various experiments on MonLink were conducted in this section to test its functionalities and performance, under different software-defined energy internet topologies managed by SDN controllers: a simple ring topology, a mesh topology and a fat-tree topology, as shown in Figure 5a–c. Hosts (e.g., h1, h2, etc.) in these figures represent power grid specific nodes such as battery sensors, surveillance cameras, etc., that generated energy internet-related data forwarded by MonLink-compatible OVS switches (e.g., s1, s2, etc.) controlled by the MonLink-enabled SDN controller. The SDN controller monitored the real-time topology and status information of all MonLink-compatible switches. MonLink was compared with sFlow (see details in Section 4.2), an excellent network monitoring solution, in these experiments. All topologies were deployed in Mininet [32] with Floodlight as the controller and OVS as OpenFlow switches. The experiment configuration was: Ubuntu 14.04 server with 64 GB memory, and 40 logical 1200 MHz CPUs.

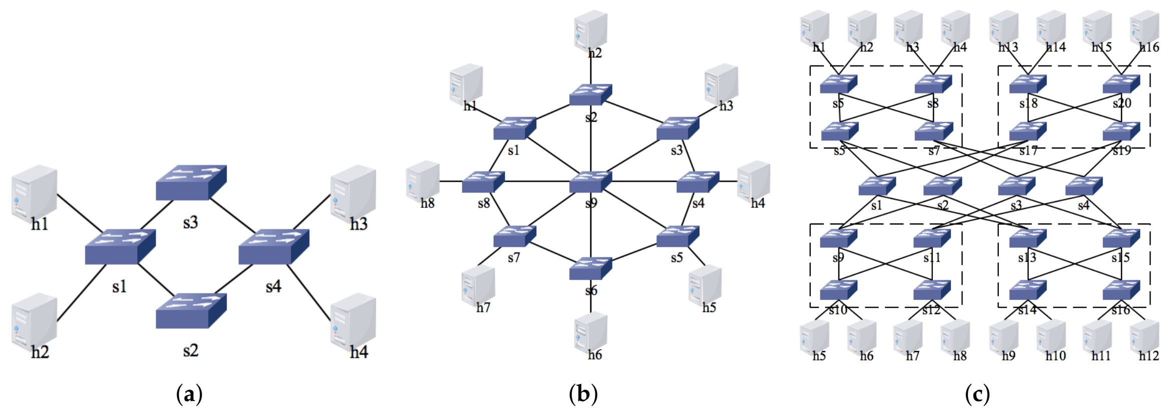


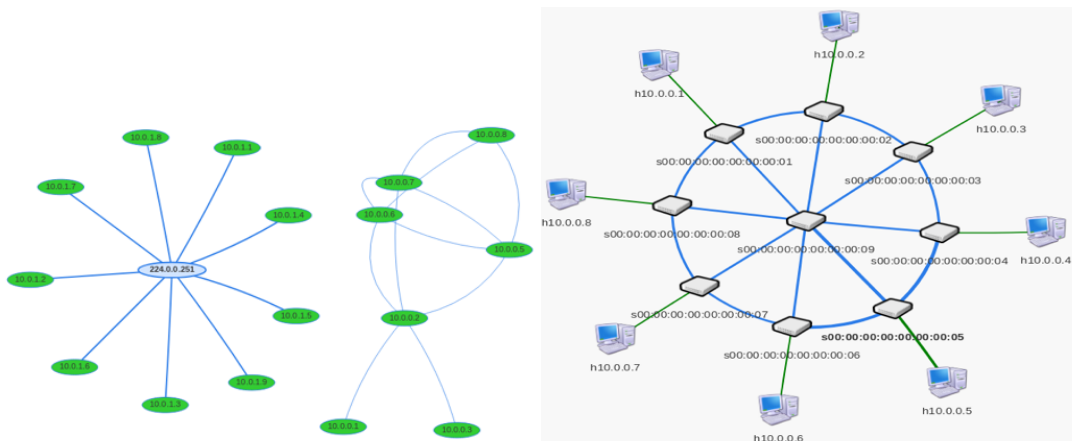
Figure 5. Topologies for experiments. (**a**) The ring topology. (**b**) The mesh topology. (**c**) The fat-tree topology.

3.1. Topology Discovery

In this section, sFlow and MonLink were compared to test their capabilities of acquiring topological information. A mesh topology (Figure 5b) was deployed for this experiment. Experimental results are shown in Figure 6, the snapshots taken from sFlow and MonLink Web GUIs, respectively.

The sFlow, as introduced above, was a network monitoring system, which also provided topology discovery functionality. However, the topological information acquisition was inaccurate as shown in Figure 6a, which is quite different from the deployed mesh topology. The result indicates that sFlow's topology discovery did not work well in SDN environments compared with the SDN-native LLDP topology discovery. As mentioned earlier, it is a desirable feature in energy internet if smart power equipments can be discovered, monitored and controlled by a smart central hub once they are installed possibly through some automated mechanics. Nevertheless, the inaccuracy in sFlow-based topology auto-discovery prevented its application in energy internet. The result also implied that if the controller wanted to acquire both topological information and status information by means of sFlow, sFlow (purely for status collecting) and LLDP (purely for topology discovery) needed to be deployed alongside and collaborate with each other. This, as a consequence, put more deployment and administrative burden on network administrators, on the one hand; on the other hand, the status information collected by sFlow and topological information collected by LLDP needed to be carefully aggregated to generate a consistent status-aware topology view identical to real topology, which is a challenging task that required plentiful programming efforts from controller/application developers. The misplacement of status information collected by sFlow on network elements/links discovered by LLDP would cause serious bad decision-making for status-sensitive applications.

MonLink, as explained above, is an extended protocol based on LLDP. It natively inherited well-performed topology discovery capability from LLDP. The experiment result indicated the topology discovered by MonLink is identical (Figure 6b) to real topology as shown in Figure 5b, plus the lightweight status information collecting (see Section 3.2).



(**a**) sFlow (**b**) MonLink

Figure 6. Topology discovery: sFlow v.s. MonLink.

3.2. *Status Information Monitoring Performance*

The sFlow and MonLink are compared in terms of runtime traffic overhead during status information collecting under different topologies (Figure 5a–c) and traffic settings (with or without upper layer video streaming traffic to mimic camera surveillance in energy internet). Three different schemes are tested in this section:

- Pure LLDP: LLDP merely conducted topology discovery without status information by means of LLDPDU multicasts and packet_in/out interactions. There packets can be considered as the benchmark overhead traffic for basic topology discovery. Therefore, LLDP was used as the benchmark to see how much more overhead the other two had to introduce in order to fulfil status information collecting in addition to basic topology discovery. This scheme used only one protocol, i.e., LLDP.
- MonLink: the extended LLDP collected status information in a piggyback fashion during LLDP-based topology discovery, thus a status-aware topology discovery. Only one extended protocol (MonLink) was used in this scheme. A modified Floodlight and OVS with support to MonLink were deployed to construct the experimental software-defined energy internet environment.
- sFlow + LLDP: LLDP conducted topology discovery while sFlow collected status information. Two protocols (sFlow and LLDP) were used in this scheme to fulfil the status-aware topology discovery in order to fairly compare with MonLink. Also, sFlow network monitoring system was deployed.

Since the default sampling interval of pure LLDP was 15 s, we set the sampling intervals of MonLink and sFlow as 15 s as well for fair comparison. The experiment results are shown in Figure 7e–f. Yellow columns exhibit the traffic overhead introduced by pure LLDP where solid-filled columns indicate the overhead ratio to the full traffic in terms of packets and pattern-filled columns indicate the overhead ratio to the full traffic in terms of bytes. Blue columns (both solid-filled and pattern-filled) indicate traffic overhead by MonLink while green columns indicate that by sFlow + LLDP. The x-axis indicates experiment time, namely 10, 20 and 30 min. Figure 7a–c are experiment results without video streaming while Figure 7d–f indicate results with video streaming from the server host, hosting a

3.2 GB mp4 file, to the client host, to mimic camera surveillance through IP-based streaming [33] in energy internets. Traffic was captured and analyzed using Wireshark.

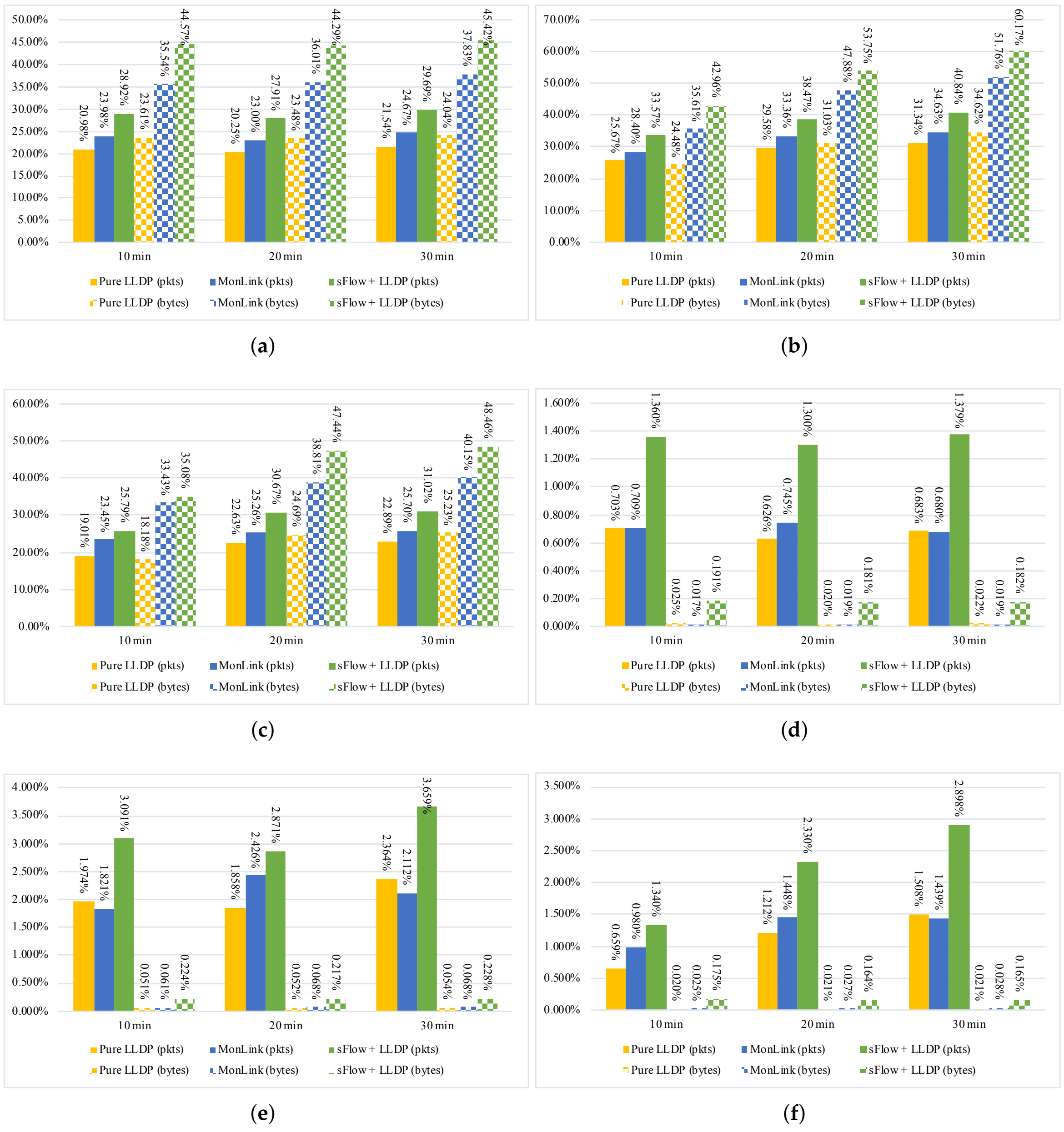


Figure 7. Traffic overhead comparison: MonLink v.s. sFlow. (**a**) Traffic overhead under ring topology without user traffic. (**b**) Traffic overhead under mesh topology without user traffic. (**c**) Traffic overhead under fat-tree topology without user traffic. (**d**) Traffic overhead under ring topology with video traffic. (**e**) Traffic overhead under mesh topology with video traffic. (**f**) Traffic overhead under fat-tree topology with video traffic.

According to the experimental results, we can see that the traffic overhead caused by MonLink was slightly greater compared with the pure LLDP benchmark. Take Figure 7f 10 min as an example (i.e., traffic overhead in fat-tree topology with 10 min video streaming), MonLink traffic overhead in bytes was 0.025% while pure LLDP benchmark overhead was 0.02%, achieving status awareness during topology discovery at the cost of very little traffic overhead. Nevertheless, traffic overhead caused by sFlow status information collecting was much greater than that of pure LLDP benchmark, almost 8.75 times as much, due to much larger packet encapsulation (up to 592 bytes). Similar analysis applies to Figure 7a–e.

To summarize, MonLink provided desirable status information monitoring performance with very low traffic overhead due to its piggyback methodology, thus a lightweight status information monitoring infrastructure for various status-aware applications.

4. Discussion

4.1. LLDP Extensions

In SDN arena, LLDP is the primary protocol used for topology discovery as mentioned in Section 1, widely adopted by various controllers such as Floodlight [31], ODL (OpenDaylight) [34], ONOS (Open Network Operating System) [35], Ryu [36], POX, NOX, etc. Therefore, extensions to LLDP are derived to achieve various purposes. Reference [37] proposed the LLDP-based latency monitoring scheme using the time-stamped LLDP packets to reduce control plane overhead and enhance measurement accuracy. Reference [38] implemented the tracing of hardware addresses in layer two bridged networks by providing self-defined TLVs. Reference [39] improved the topology discovery by LLDP, and proved that shortest paths can be built at the same time that the topology information is gathered, thus an enhanced topology discovery service for SDN. Note that some topology detection and forensics [40] techniques are also developed based on LLDP from the perspective of network security, which can also serve for the purpose of topology discovery.

These schemes share the similar thought with our scheme that implementing extra control plane functionalities based on LLDP does not lead to remarkable overhead and information can be piggybacked in self-defined fields. Our scheme differs from these works in that we derived various LLDP working modes that provide both self-adaptiveness and high interactiveness (by means of REST APIs, i.e., Representational State Transfer Application Programming Interfaces, at northbound interfaces) for upper layer applications. In addition, our work presents an architecturally better systematic perspective that the piggybacked LLDP-based information collecting and monitoring scheme can serve as a common underlying infrastructure for status-awareness by providing application-independent and platform-agnostic APIs. In this way, upper layer applications can focus on the business logics instead of implementing ad hoc information acquisition mechanisms.

4.2. Network Monitoring

Status information collecting can be done based on network monitoring, among which, sFlow, developed by InMon etc., is a typical representative. sFlow works by means of random traffic sampling. It provides traffic statistics from layer 2–4, enabling the real-time analysis on performance and trending as well as trouble shooting. sFlow consists of agents and collectors. Agents, as clients, are usually embedded in network devices such as switches, routers, etc., sending collected information encapsulated as sFlow packets (RFC 3176) [24] to the collector (i.e., the server) which generates traffic reports based on received information.

However, status information collecting based on sFlow in SDN requires the deployment of sFlow, with proper configuration and integration into SDN networks. This results in heavy administrative and runtime burden. Meanwhile, sFlow does not provide highly accurate topology discovery (see Section 3.1), thus requiring the collaboration with native LLDP topology discovery process. Besides, status information collected by sFlow and topological information collected by LLDP must be carefully aggregated to construct a consistent status-aware topological view. Last but not least, each sFlow packet contains 592 bytes by default, which would lead to remarkable traffic overhead. Our scheme overcomes problems seen in the sFlow-based scheme in that small sized metric TLVs collect status information during the topology discovery (See Section 3).

4.3. The Integration of SDN into Energy Internet

Recent years have also seen the integration of SDN into energy internet [1]/smart grid [41]. Early works originate mainly from industry (power grid enterprises). Reference [42] proposes the conceptual efficiency evaluation index system of complex power grid based on SDN and analytic hierarchy process (AHP). Similar work is proposed in reference [43]. However, these works are still at very early stage, in urgent need for theoretical depth and application soundness.

Recent efforts are also seen in academia. Realizing that inaccurate synchronization may lead to maliciously fluctuations for energy data monitoring, reference [41] proposes a software-defined on-path time synchronization (SD-OPTS) scheme for supervisory control and data sensing in smart grid to increase the flexibility, controllability and reliability of time synchronization. Our scheme differs in that we focus on status monitoring itself in energy internet while their work studied mechanisms that help with monitoring. Reference [44] proposes the wireless distribution network (WDN) as opposed to traditionally wired PDN. WDN is designed with SDN concept where SDN controller acts as the network manager controlling multiple agents deployed throughout the WDN. It proves the feasibility of SDN-governed wireless technologies in the field of smart grid. However, the QoS provisioning is not considered in this paper, which requires extensive status monitoring like the scheme we proposed in this paper. Reference [45] proposes the cooperative mechanism for energy transportation and storage in energy internet. The main purpose of the research is to well-utilize the redundant energy of a given Power Distribution Network, so that it is not wasted as treated in traditional PDN. It is modeled as an NP-hard Knapsack problem and heuristics are derived to solve this problem. To achieve this, a complementary SDN platform is constructed dedicated for decision making. Statistics for decision making are collected by means of flow table entries pushed from power routers to controllers, contradictory to the common SDN workflow that controllers push flow tables to switches/routers. Our scheme follows and extends the standard SDN workflow procedures thus exhibiting good compatibility.

5. Conclusions

Since LLDP is the mainstream method for topology discovery in SDN arena, extra control plane functionalities can hitchhike LLDP to achieve traffic efficiency without introducing newly designed dedicated protocol. In this paper, we proposed MonLink, a piggyback status information collecting scheme based on LLDP in software-defined energy internet. The extended metric TLVs enable the status information collecting during topology discovery. Several working modes—periodic, adaptive and proactive MonLink—are derived at southbound interface to achieve flexible collecting. Various REST APIs are implemented at northbound interface for easier upper-application invocations. Compared with traditional method based on sFlow, MonLink is a lightweight status information collecting framework. Meanwhile the hitchhiking on LLDP ensures that MonLink does not compromise or affect the normal workflow at southbound interface. In our future work, we are going to study the possibility of integrating MonLink with machine learning to improve the adaptive MonLink with adaptive status benchmark settings.

Author Contributions: Conceptualization, X.C.; methodology, X.C., S.L. and H.Y.; software, X.C., S.L.; validation, X.C., H.Y. and A.K.S.; formal analysis, A.K.S.; investigation, X.C. and Y.C.; resources, A.K.S.; data curation, X.C. and A.K.S.; writing—original draft preparation, X.C.; writing—review and editing, X.C., H.Y. and A.K.S.; visualization, X.C., Y.C. and S.L.; supervision, H.Y.; project administration, H.Y.; funding acquisition, X.C. and H.Y.

Funding: This research was funded by the China Postdoctoral Science Foundation (Grant No. 2018M643448), Sichuan Science and Technology Program (Grant No. 2019YFG0110) and Fundamental Research Funds for the Central Universities, Southwest Minzu University (Grant No. 2019NQN22).

Acknowledgments: The authors also thank the time and efforts by the reviewers and editors during the reviewing process.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Rifkin, J. *The Third Industrial Revolution: How Lateral Power Is Transforming Energy, the Economy, and the World*; St. Martin's Press: New York, NY, USA, 2011.
2. Crow, M.L.; McMillin, B.; Wang, W.; Bhattacharyya, S. Intelligent Energy Management of the FREEDM System. In Proceedings of the IEEE PES General Meeting, Providence, RI, USA, 25–29 July 2010; pp. 1–4.
3. Huang, A.Q.; Crow, M.L.; Heydt, G.T.; Zheng, J.P.; Dale, S.J. The Future Renewable Electric Energy Delivery and Management (FREEDM) System: The Energy Internet. *Proc. IEEE* **2011**, *99*, 133–148. [CrossRef]
4. Schmid, J.; Strauss, P.; Hatziargyriou, N.; Akkermans, H.; Buchholz, B.; Van Oostvoorn, F.; Scheepers, M.; Reyero, R.; Chadjivassiliadis, J. *Towards Smart Power Networks: Lessons Learned From European Research FP5 Projects*; European Commission Directorate-General for Research Information and Communication Unit: Brussels, Belgium, 2005.
5. Kreutz, D.; Ramos, F.M.V.; Esteves Verissimo, P.; Esteve Rothenberg, C.; Azodolmolky, S.; Uhlig, S. Software-Defined Networking: A Comprehensive Survey. *Proc. IEEE* **2015**, *103*, 14–76. [CrossRef]
6. Soliman, M.; Nandy, B.; Lambadaris, I.; Ashwood-Smith, P. Exploring source routed forwarding in SDN-based WANs. In Proceedings of the IEEE International Conference on Communications (ICC), Sydney, NSW, Australia, 10–14 June 2014; pp. 3070–3075.
7. Sun, G.; Chang, V.; Yang, G.; Liao, D. The cost-efficient deployment of replica servers in virtual content distribution networks for data fusion. *Inf. Sci.* **2018**, *432*, 495–515. [CrossRef]
8. Miao, W.; Agraz, F.; Peng, S.; Spadaro, S.; Bernini, G.; Perelló, J.; Zervas, G.; Nejabati, R.; Ciulli, N.; Simeonidou, D.; et al. SDN-enabled OPS with QoS guarantee for reconfigurable virtual data center networks. *IEEE/OSA J. Opt. Commun. Netw.* **2015**, *7*, 634–643. [CrossRef]
9. Sun, G.; Liao, D.; Zhao, D.; Xu, Z.; Yu, H. Live Migration for Multiple Correlated Virtual Machines in Cloud-Based Data Centers. *IEEE Trans. Serv. Comput.* **2018**, *11*, 279–291. [CrossRef]
10. Sun, G.; Anand, V.; Liao, D.; Lu, C.; Zhang, X.; Bao, N.H. Power-efficient provisioning for online virtual network requests in cloud-based data centers. *IEEE Syst. J.* **2015**, *9*, 427–441. [CrossRef]
11. Sun, G.; Liao, D.; Zhao, D.; Sun, Z.; Chang, V. Towards provisioning hybrid virtual networks in federated cloud data centers. *Future Gen. Comput. Syst.* **2018**, *87*, 457–469. [CrossRef]
12. Wang, L.; Lu, Z.; Wen, X.; Knopp, R.; Gupta, R. Joint Optimization of Service Function Chaining and Resource Allocation in Network Function Virtualization. *IEEE Access* **2016**, *4*, 8084–8094. [CrossRef]
13. Yu, H.; Wen, T.; Di, H.; Anand, V.; Li, L. Cost efficient virtual network mapping across multiple domains with joint intra-domain and inter-domain mapping. *Opt. Switch. Netw.* **2014**, *14*, 233–240. [CrossRef]
14. Yu, H.; Qiao, C.; Wang, J.; Li, L.; Anand, V.; Wu, B. Regional failure-resilient virtual infrastructure mapping in a federated computing and networking system. *IEEE/OSA J. Opt. Commun. Netw.* **2014**, *6*, 997–1007. [CrossRef]
15. Yang, K.; Zhang, H.; Hong, P. Energy-Aware Service Function Placement for Service Function Chaining in Data Centers. In Proceedings of the 2016 IEEE Global Communications Conference (GLOBECOM), Washington, DC, USA, 4–8 December 2016; pp. 1–6.
16. Wen, T.; Yu, H.; Du, X. Performance guarantee aware orchestration for service function chains with elastic demands. In Proceedings of the 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), Berlin, Germany, 6–8 November 2017; pp. 1–4.
17. Sun, G.; Li, Y.; Liao, D.; Chang, V. Service function chain orchestration across multiple domains: A full mesh aggregation approach. *IEEE Trans. Netw. Serv. Manag.* **2018**, *15*, 1175–1191. [CrossRef]
18. Sun, G.; Li, Y.; Yu, H.; Vasilakos, A.V.; Du, X.; Guizani, M. Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks. *Future Gen. Comput. Syst.* **2019**, *91*, 347–360. [CrossRef]
19. Sun, G.; Zhu, G.; Liao, D.; Yu, H.; Du, X.; Guizani, M. Cost-Efficient Service Function Chain Orchestration for Low-Latency Applications in NFV Networks. *IEEE Syst. J.* **2018**, 1–13. [CrossRef]
20. Sun, G.; Li, Y.; Li, Y.; Liao, D.; Chang, V. Low-latency orchestration for workflow-oriented service function chain in edge computing. *Future Gen. Comput. Syst.* **2018**, *85*, 116–128. [CrossRef]

21. Mckeown, N.; Anderson, T.; Balakrishnan, H.; Parulkar, G.; Peterson, L.; Rexford, J.; Shenker, S.; Turner, J. OpenFlow: Enabling innovation in campus networks. *Acm Sigcomm Comput. Commun. Rev.* **2008**, *38*, 69–74. [CrossRef]
22. ONF. OpenFlow Switch Specification Ver 1.3.5. 2015. Available online: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.3.5.pdf (accessed on 3 January 2019).
23. IEEE Standard for Local and Metropolitan Area Networks—Station and Media Access Control Connectivity Discovery. *IEEE Std 802.1AB-2016 (Revision of IEEE Std 802.1AB-2009)*; IEEE: Piscataway, NJ, USA, 2016; pp. 1–146.
24. Panchen, S.; McKee, N.; Phaal, P. *InMon Corporation's sFlow: A Method for Monitoring Traffic in Switched and Routed Networks*; Number 3176 in Request for Comments, RFC, Ed.; RFC 3176; IETF: Fremont, CA, USA, 2001.
25. Pfaff, B.; Pettit, J.; Koponen, T.; Jackson, E.; Zhou, A.; Rajahalme, J.; Gross, J.; Wang, A.; Stringer, J.; Shelar, P. The design and implementation of open vSwitch. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*; USENIX Association: Oakland, CA, USA, 2015; Volume 40, pp. 117–130.
26. Zhou, Z.; Zhang, H.; Du, X.; Li, P.; Yu, X. Prometheus: Privacy-aware data retrieval on hybrid cloud. In Proceedings of the 2013 Proceedings IEEE INFOCOM, Turin, Italy, 14–19 April 2013; pp. 2643–2651.
27. Cheng, Y.; Fu, X.; Du, X.; Luo, B.; Guizani, M. A lightweight live memory forensic approach based on hardware virtualization. *Inf. Sci.* **2017**, *379*, 23–41. [CrossRef]
28. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 281–305.
29. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; Riedmiller, M. Playing Atari with Deep Reinforcement Learning. *arXiv* **2013**. arXiv: 1312.5602.
30. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous Control with Deep Reinforcement Learning. *arXiv* **2015**. arXiv: 1509.02971.
31. Floodlight, S. OpenFlow Controller. Available online: https://github.com/floodlight/floodlight (accessed on 3 January 2019).
32. Lantz, B.; Heller, B.; McKeown, N. A Network in a Laptop: Rapid Prototyping for Software-defined Networks. In Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks, Monterey, CA, USA, 20–21 October 2010; Hotnets-IX, pp. 19:1–19:6.
33. Xiao, Y.; Du, X.; Zhang, J.; Hu, F.; Guizani, S. Internet protocol television (IPTV): The killer application for the next-generation internet. *IEEE Commun. Mag.* **2007**, *45*, 126–134. [CrossRef]
34. Medved, J.; Varga, R.; Tkacik, A.; Gray, K. OpenDaylight: Towards a Model-Driven SDN Controller architecture. In Proceedings of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014, Sydney, NSW, Australia, 19 June 2014; pp. 1–6.
35. Berde, P.; Gerola, M.; Hart, J.; Higuchi, Y.; Kobayashi, M.; Koide, T.; Lantz, B.; O'Connor, B.; Radoslavov, P.; Snow, W.; et al. ONOS: Towards an Open, Distributed SDN OS. In Proceedings of the ACM Third Workshop on Hot Topics in Software Defined Networking, Chicago, IL, USA, 17–22 August 2014; ACM: New York, NY, USA, 2014; pp. 1–6.
36. Kubo, R.; Fujita, T.; Agawa, Y.; Suzuki, H. Ryu SDN Framework—Open-Source SDN Platform Software. *NTT Tech. Rev.* **2014**, *12*, 1–5.
37. Liao, L.; Leung, V.C.M. LLDP based link latency monitoring in software defined networks. In Proceedings of the 2016 12th International Conference on Network and Service Management (CNSM), Montreal, QC, Canada, 31 October–4 November 2016; pp. 330–335.
38. Scholz, T.; Kruse, J. Tracing of hardware addresses in layer two bridged networks. In Proceedings of the 2015 IEEE International Conference on Communications (ICC), London, UK, 8–12 June 2015; pp. 1025–1030.
39. Rojas, E.; Alvarez-Horcajo, J.; Martinez-Yelmo, I.; Carral, J.A.; Arco, J.M. TEDP: An Enhanced Topology Discovery Service for Software-Defined Networking. *IEEE Commun. Lett.* **2018**, *22*, 1540–1543. [CrossRef]
40. Achleitner, S.; Porta, T.L.; Jaeger, T.; Mcdaniel, P. Adversarial Network Forensics in Software Defined Networking. In Proceedings of the Symposium on SDN Research, Santa Clara, CA, USA, 3–4 April 2017; pp. 8–20.

41. Han, W.; Dong, M.; Ota, K.; Wu, J.; Li, J.; Li, G. SD-OPTS: Software-Defined On-Path Time Synchronization for Information-Centric Smart Grid. In Proceedings of the GLOBECOM 2017-2017 IEEE Global Communications Conference, Singapore, 4–8 December 2017; pp. 1–6.
42. Zhang, Q.; Wang, H.; Song, Y. Efficiency evaluation algorithm of SDN for energy internet. In Proceedings of the 2017 7th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC), Macau, China, 21–23 July 2017; pp. 292–295.
43. Zhang, G.; Su, L.; Wang, Y.; Liu, X.; Li, J. Research on communication network architecture of energy internet based on SDN. In Proceedings of the 2014 IEEE Workshop on Advanced Research and Technology in Industry Applications (WARTIA), Ottawa, ON, Canada, 29–30 September 2014; pp. 316–319.
44. Li, B.; Kong, L.; Cao, W.; Yang, Z.; We, L. A novel wireless distribution network application to support further Internet of Energy. In Proceedings of the 11th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM 2015), Shanghai, China, 21–23 September 2015; pp. 1–6.
45. Hou, W.; Tian, G.; Guo, L.; Wang, X.; Zhang, X.; Ning, Z. Cooperative Mechanism for Energy Transportation and Storage in Internet of Energy. *IEEE Access* **2017**, *5*, 1363–1375. [CrossRef]