Secure Scalable VPLS: A Lagrangian Relaxation Approach to Tunnel Relaying Optimization

Mohammad Borhani*, Ioannis Avgouleas†, Madhusanka Liyanage‡, Andrei Gurtov*
* Dept. of Computer and Information Science, Linköping University, Sweden
†Ericsson AB, Sweden ‡School of Computer Science, University College Dublin, Ireland

***Abstract*—Virtual Private LAN Service (VPLS) is commonly used for secure multi-point communication across geographically scattered industrial sites, simulating a unified LAN broadcast domain for Industrial IoT (IIoT)-type devices. This configuration demands a fully-connected overlay network with encrypted Host Identity Protocol (HIP)/IPsec tunnels exhibiting quadratic scalability to the number of tunnels and a significant increase in forwarding table entries. Herein, we introduce Tunnel Relay Nodes (TRNs) as selected routers that maintain full-mesh connectivity. This approach allows non-TRN routers, or Provider Equipment (PEs) acting as spoke PEs, to connect via a TRN. We explore the challenges of using TRNs in secure HIP-based VPLS (HIPLS) networks, including (i) placing reliable TRNs within provider networks and (ii) scheduling TRNs to minimize their activation/deactivation costs as well as the connection cost among PEs. We then demonstrate how (i) can be addressed in polynomial time using a modified general median problem approach. Additionally, we formulate (ii) as a Mixed Integer Linear Programming (MILP) scheduling problem and prove its NP-completeness. Furthermore, we introduce an algorithm based on Lagrangian relaxation to address the intractability in large-scale deployments. This algorithm offers fast, near-optimal solutions while simultaneously balancing solution quality and execution time. Our simulations on three real-world network topologies with real network demands show a 92% average reduction in forwarding table entries on PE. Compared to existing solutions, our method reduces the number of tunnels established by up to 95%, at the expense of a 1.39-fold increase in tunnel path length.**

***Index Terms*—IIoT, VPLS, HIPLS, Network Design, Optimization**

I. Introduction

A. Motivation

The interconnection of Industrial IoT (IIoT) networks is essential for modern industries seeking optimal operational efficiency, instantaneous data handling, remote control, and enhanced decision-making, particularly in widely distributed regions. Industries such as oil and gas, mining, and agriculture generally operate in extensive, isolated regions where conventional on-site monitoring and management techniques are ineffective and expensive [1]–[3]. In addition, the IIoT market was valued at 77 billion dollars in 2021 and is estimated to expand to 106 billion dollars by 2026, demonstrating a compound annual growth rate (CAGR) of 6% [4].

Cyberattacks on industrial systems have escalated significantly, largely due to increased remote access capabilities, inadequate security designs in IIoT devices, and the expanded attack surface resulting from enhanced Internet connectivity. This augmentation in cyber vulnerabilities not only undermines the functionality of industrial systems but also, in critical scenarios, poses a considerable risk to human safety [5], [6]. Historically, industrial environments have been susceptible to significant cyberattacks. Notable examples include the 2021 cyberattacks on the US energy sector, the Triton incident in 2017, the Black Energy attack in 2015, and the Stuxnet virus in 2010 [7]. According to the Kaspersky ICS CERT reports [8], [9], during the initial six months of 2023, approximately 36% of the Industrial Control System (ICS) computers were subjected to at least one cyberattack. The report suggests a continued trend, predicting that ransomware will remain the predominant threat to industrial enterprises in the future [8], [9].

The presence of industrial device scanners such as Shodan and Nmap emphasizes the urgent need to protect vulnerable industrial devices from attacks. Furthermore, the COVID-19 pandemic's acceleration of remote work has made it imperative to prioritize strategies that conceal industrial devices from exposure to the Internet while allowing remote access for monitoring and control of industrial activities [10], [11].

Virtual Private Networks (VPNs) establish private networks within the public infrastructure by delivering encrypted, secure connections over the Internet. This protects sensitive information transmitted to and from industrial devices against unauthorized access [12]. Among the diverse range of VPN technologies, Virtual Private LAN Service (VPLS) stands out as a frequently employed alternative for site-to-site connectivity. Characterized as a Layer 2 VPN administered by service providers, VPLS is adept at delivering Ethernet connectivity in a multi-point to multi-point configuration, utilizing either IP or MPLS networks. This enables the extension of Local Area Networks (LANs) across diverse customer locations (industrial sites), effectively merging them into a single LAN broadcast domain. This attribute renders VPLS an apt solution for IIoT devices operating at Layer 2 [13]–[15]. Nevertheless, geographical distance introduces latency that practically limits the feasible reach of a VPLS, especially in latency-sensitive industrial applications.

By establishing a shared Layer 2 domain, VPLS offers several advantages, including:

- Operational cost-efficiency: VPLS utilizes existing underlay infrastructure to minimize the need for additional

This work was supported in part by the Graduate School in Computer Science (CUGS), and in part by the Excellence Center at Linköping–Lund in Information Technology (ELLIIT) A.4 project.

dedicated lines, thereby reducing deployment and maintenance costs.
- Homogeneous IIoT networks integration: VPLS enables the integration of multiple industrial sites of IIoT devices into a single, homogeneous network.
- Rapid deployment: The ability to leverage existing infrastructure, coupled with the inherent feature of easily adding more sites, facilitates the rapid establishment of isolated networks of IIoT sites.

The increasing popularity of the IIoT requires strict security measures, enhanced scalability, and efficient utilization of network resources. Moreover, the escalating number of IIoT devices and new application services underscores the need for secure IIoT communication. The Host Identity Protocol (HIP) separates endpoint identifiers from locators, enhancing security through identity verification while offering basic security services such as encryption, confidentiality, and data integrity. After exchanging encryption keys and authenticating the peers by HIP base exchange (BEX), HIP shields against identity spoofing, facilitates encrypted communication using an IPsec ESP tunnel, and ensures data authenticity. Its architecture inherently obscures device IP (IPv4 or IPv6 addresses), offering significant protection against DDoS attacks, crucial for maintaining continuous, secure network operations in the IIoT environment. HIP has been effectively incorporated into VPLS, endowing VPLS with its comprehensive security capabilities [16]–[18]. HIP-based VPLS (HIPLS) merges HIP's robust security features with VPLS, fostering secure, identity-based VPLS [16]. VPLS networks utilizing HIP are widely employed across various industries such as healthcare, manufacturing, and transportation, including their application in Boeing's 777 airplane assembly.

While HIPLS presents a promising solution for IIoT connectivity, its scalability challenges in the control plane significantly limit its applicability. Issues (which will be thoroughly discussed in Section III-A) such as the necessity for full-mesh tunneling across interconnected customer sites and the extensive expansion of Provider Edge (PE) forwarding table entries hinder its efficiency in IIoT large-scale deployment. Moreover, these limitations contribute to elevated deployment costs for service providers [12].

B. Contributions

To address the management and scalability issues in a secure VPLS network, we use the concept of relaying. The relaying approach selects a small subset of routers as relay nodes while maintaining full-mesh reachability to all other routers in the network, thus reducing the number of tunnels established. This allows non-relay nodes to reach other nodes by relaying through a specified relay node.

To the best of our knowledge, this is the first study on the placement of reliable relays, the optimal selection of relays for routers, and the activation and deactivation of relays based on network dynamics in secure HIPLS. The main contributions of this paper are:

- We identify and motivate the Tunnel Relay Node (TRN) problem in HIPLS networks by analyzing three key limitations hindering large-scale IoT deployment: N-square scalability, PE forwarding table expansion, and static tunnel time.
- We formulate the TRN placement as a variant of the general median problem on graphs and extend it to include failure resistance through the K-tunnel Relay Placement Problem. We propose a polynomial-time algorithm to compute median-based placements of TRNs.
- We introduce the Tunnel Scheduling (TS) problem as a Mixed Integer Linear Program (MILP) to optimize TRN activation based on network demand, accounting for cost and capacity constraints. (TS) also determines the association between PEs and active TRNs. We prove (TS) is NP-complete and develop an efficient Lagrangian relaxation algorithm (including a feasible solution generation algorithm) to attack the complexity.
- Our performance evaluations on real-world network topologies under actual demands demonstrate that the proposed methods yield near-optimal solutions for small and large scenarios (validated via exact methods). Moreover, they outperform state-of-the-art solutions in terms of tunnel count, complexity, and forwarding-table requirements, thereby enabling practical, large-scale IIoT deployments, while introducing only a minor, bounded increase in end-to-end path length.

C. Article Organization

The remainder of this paper is organized as follows: Section II provides a brief background on HIPLS. Section III presents an overview of the TRN problem, the mathematical problem formulation for placing reliable relay nodes, and the (TS) problem for tunnel scheduling. The proposed solutions to the (TS) problem are detailed in Section IV. Section V discusses the simulation results. Related work is discussed in Section VI. Finally, Section VII concludes the paper.

II. Background

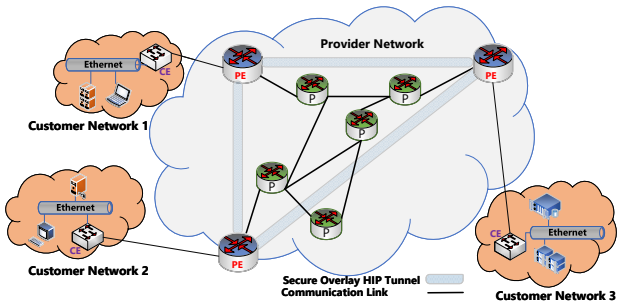

Figure 1. HIPLS architecture.

This section presents an overview of HIPLS. As illustrated in Figure 1, HIPLS consists of several crucial elements, including:

- **Customer Sites (Networks)**: Customer networks refer to the sites within a VPLS network managed by the end users. These sites are spread across various geographic locations.

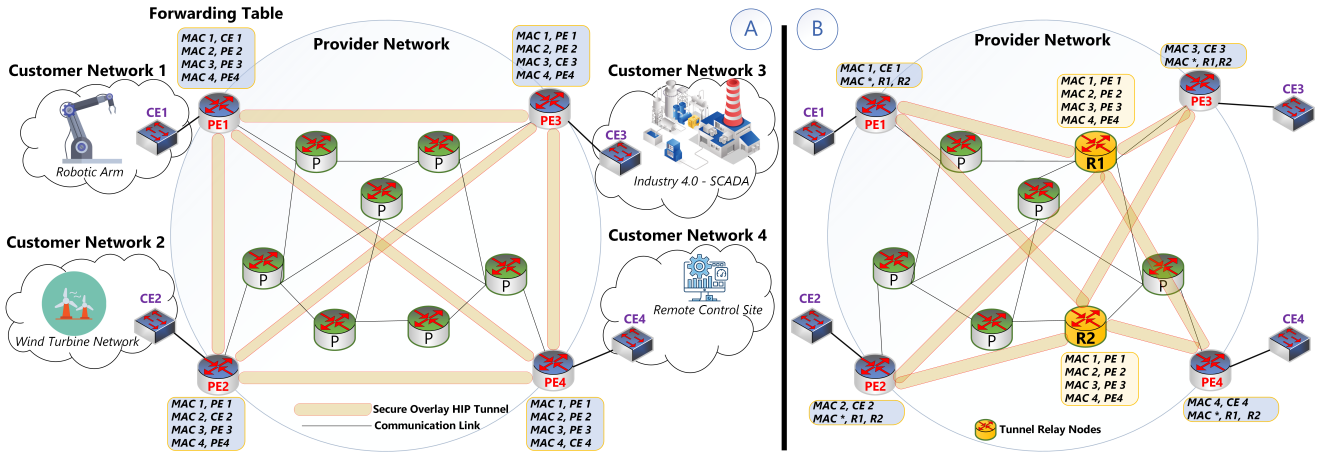

Figure 2. PE Memory footprint and required HIP tunnels for secure HIPLS network: (A) full-mesh and (B) Tunnel Relay Node (proposed solution). In the latter, only the Tunnel Relay Node PEs are fully meshed with all PEs, reducing forwarding table entries for spoke PEs and breaking the quadratic number of required tunnels for large-scale networks at the cost of a minor tunnel path length increase.

- **Provider Edge (PE) routers**: PEs serve as gateway routers in customer networks, functioning similarly to learning bridges for customer sites, and are situated at the boundary of the provider network. PEs are fully aware of the VPLS. To facilitate secure network connectivity, tunnels are established between PEs.
- **Customer Edge (CE) equipment**: CEs, typically owned by the customer, are located on the customer's premises. A CE might be connected to one or more PE routers. CEs can encompass a variety of network equipment, such as hosts, routers, bridges, or switches, and may operate at L2/L3.
- **Provider routers (P routers)**: P routers are transit routers within the provider's network infrastructure. P router operates without knowing the VPLS. Its primary function is to forward traffic within the provider's network.
- **HIP tunnels**: A VPN tunnel represents an encrypted/encapsulated connection established between two PEs. These tunnels are typically created using IPsec Encapsulating Security Payload (ESP), with their management governed by the HIP signaling protocol. Because IPsec Security Associations (SAs) are typically one-way, two SAs (in BEET ESP mode) are needed per connection for full-duplex traffic, effectively doubling the tunnel entries.

III. Problem Formulation

A. Problem Overview

Full-mesh and multipoint-to-multipoint reachability of HIPLS result in two major costs in the provider network: (i) N-square scalability problem with respect to the number of tunnels between PEs for large deployments, (ii) PE forwarding table entries expansion.

N-Square scalability problem. The small-scale secure VPLS (i.e., HIPLS network) is detailed in Figure 2. Secure connectivity between geographically dispersed customer sites is essential for their operations (e.g., the operator at customer site 4 must be able to remotely transmit commands to the robotic arm at customer site 1). Using an underlay IP/MPLS-based network, the provider establishes a full-mesh of overlay HIP tunnels between all PEs.

In the full-mesh HIPLS architecture, each PE device establishes direct connections with every other PE device in the network via HIP tunnels. While full-mesh reachability offers benefits such as reduced latency between PEs, it also introduces drawbacks, including:

- *Scalability problem*: the total amount of required HIP tunnels escalates rapidly as PE routers increase. To be more precise, assuming that n represents the number of PEs in the network, the total number of tunnels needed is $\frac{n(n-1)}{2}$. This may result in significant scalability challenges, as each PE manages many tunnels and the corresponding signaling.
- *Complexity and Resource Intensive*: The management of the full-mesh reachability becomes progressively more intricate as the number of PEs increases. Maintaining and configuring a significant number of tunnels can present difficulties. Furthermore, it is costly for every PE device to monitor all tunnels connecting to other PEs within the network. This includes the establishment of HIP BEX and the management/rekeying of every secure HIP tunnel.

PE forwarding table entries expansion. Each PE router maintains a forwarding table (database) for every VPLS instance to learn the MAC addresses of the traffic arriving on the ports facing the CEs it supports. Traffic is switched based on MAC addresses, and if necessary, it is forwarded to all PEs participating in the VPLS instance. In this manner, each PE (for each VPLS instance) populates the MAC addresses from all directly connected CEs into its VPLS forwarding table, as shown by the forwarding table for PEs in Figure 2. Moreover, as network providers delivering secure VPLS should support hundreds to thousands of customers (CEs) in real-world applications, implementing this operational structure causes a rapid rise in the VPLS forwarding tables for each PE.

For example, a sizable organization might own as many as 500 office branches seeking to share the same L2 broadcast

Table I
COMPARISON OF EXISTING VPLS-BASED SOLUTIONS AND THE PROPOSED TRN ARCHITECTURE FOR IIOT.

Approach	Security Mechanism	Tunnel Scaling	Forwarding Table Growth	Implementation Complexity	IIoT Suitability
BGP/LDP VPLS [19]–[21]	No built-in security	Full-mesh (scales poorly - N square issue)	High (must learn all PEs)	Low (standard in large provider networks)	Low — not designed to address specialized IIoT constraints
HIPLS (IETF Draft) [16], [22]	End-to-end security with HIP and IPsec	Full-mesh	High (each PE learns all others)	Moderate–High (HIP configuration overhead)	Medium — provides sufficient security for limited IIoT setups, but lacks scalability
S-HIPLS [23]	HIP-based, more efficient key exchange	Full-mesh	High (same as HIPLS)	Moderate–High	Medium — minor key-exchange improvements for larger deployments
SDN-VPLS [13], [24]	Centralized (SDN) approach controlling tunnels	Full-mesh (SDN reduced overhead)	Medium (controller can optimize)	High (needs SDN controller and integration)	Medium–High — must handle new SDN stack in industrial setting
Proposed TRN-based	HIP-based overlay + TRN relay scheduling	Linear in practice	Low-moderate (fewer direct PEs)	Moderate (standard VPLS + TRN nodes)	High — addresses large-scale IIoT with efficient resource use

domain that requires interconnection. Each PE is required to maintain a minimum of 2.5K entries in its forwarding table to ensure connectivity to all sites in the VPLS instance it supports, under the extremely restrictive assumption that each branch announces five devices to communicate. Additionally, for rapid address lookup, a PE router has a restricted amount of content-addressable memory (CAM). A moderate-level PE can insert approximately 64K entries in the forwarding table for fast lookup. Clearly, utilizing (2.5K/64K = 3.9%) of the PE's total memory for a solitary VPLS instance belonging to a single customer challenges the scalability of VPLS. When 24 customers utilize the identical configuration as our example, the CAM can be easily filled, leaving no capacity for the operation of other PE services. In practice, the CAM of PE is expensive and often limited to just a few megabytes, further constraining the number of forwarding entries.

Since limited CAM restricts the number of supported CEs, maintaining a full-mesh of all-to-all tunnels is impractical for large-scale deployments. Furthermore, it is essential to note that provider routers support many other services; therefore, each PE router may be required to allocate available CAM to different services efficiently.

Tunnel Relay Nodes. One potential approach to mitigate the expansion of PE forwarding table entries and simultaneously reduce the required number of established HIP tunnels is to designate a restricted number of routers in the network as TRNs, also referred to as *hubs*. TRNs preserve the full-mesh reachability to all other PEs and allow *spoke* PEs (non-hub routers) to communicate with other PEs via the TRN. Figure 2B illustrates how spoke PEs use TRN by establishing HIP tunnels to them and reducing their VPLS forwarding table entries by installing default routes to TRN; TRN retains all reachability routes, thus, in general, significantly reducing the number of forwarding records necessary to provide the HIPLS service.

Additionally, from a practical standpoint, it is relatively uncommon for PEs to send data to every other PE in their VPLS instance very frequently. Previous research has demonstrated that site-to-site VPNs (e.g., VPLS) typically preserve sparse traffic matrices between PEs [25], [26]. Taking into account the structure of the organization and the desired services to be distributed through VPLS services for client locations, certain communication patterns can be effectively incorporated into a hub-and-spoke architecture among PEs. Table I compares the TRN-based architecture with representative VPLS-based solutions. The overview highlights key features of our TRN architecture in the context of IIoT, underscoring distinct advantages of our relay-and-scheduling design in handling large-scale, security-intensive scenarios.

Table I provides an overview of several current L2/L3 overlay solutions, contrasting their security mechanisms, scalability, and implementation complexity with our TRN-driven solution. As shown, our approach is particularly suited for large-scale IIoT deployments due to its near-linear tunnel growth and reduced forwarding overhead, while still offering robust end-to-end security. When selecting TRNs for relaying, one must make intricate decisions across various dimensions, including:

- *Reducing the number of TRN*: A smaller number of TRNs is ideal, as it causes a further decrease in PE forwarding records. Furthermore, fewer TRNs yield less CAPEX and OPEX incurred by the provider to maintain the VPLS service. Nonetheless, the capacity of each TRN to fulfill spoke PE demands is limited. Therefore, the provider must ensure that it has sufficient TRNs to meet the PE's demand.
- *Failure-resistance TRNs*: TRNs are essential for the robust operation of VPLS. Without backup plans, it is evident that a failure in TRNs could result in significant service degradation or a total halt of VPLS operations. When designing TRNs, network providers should take into account the possibility of failure and the availability of backup strategies.
- *Tunnel path stretch*: In relaying, PEs communicate via TRN, increasing the traversed path (compared to direct full-mesh communication). The tunnel path stretch ratio can be defined as the tunnel path (distance) traversed in relayed communication divided by the direct communication tunnel path traversal. Obviously, a longer tunnel path results in more delay for endpoints and more traffic to be passed over provider networks. Therefore, minimizing the tunnel path stretch and communication cost between each PE and its associated TRNs is important.
- *TRNs activation/deactivation cost*: In addition to estab-

lishing and maintaining the HIP tunnel between all PEs and TRNs, the deployment of TRNs in the provider network entails various costs, such as configuration and maintenance. Furthermore, the optimal allocation of PEs to TRNs can lead to the deactivation of certain TRNs, thereby reducing the number of HIP tunnels and provider costs, considering the dynamic demands of PEs in VPLS networks. On the contrary, deactivating TRN results in the closure of all HIP tunnels to its connecting PEs and incurs additional costs to establish the tunnels again when needed. The optimization process should consider activation and deactivation costs for each TRN.

- *TRNs security considerations:* The use of TRNs architecture, which incorporates HIP for security, ensures that the security of VPLS is not compromised compared to the full-mesh architecture. However, switching from a full-mesh VPLS design to a TRNs-based design means each PE must establish a secure HIP tunnel with its designated TRNs rather than with every other PE, as is the case in a full-mesh setup. This change places an additional load on the TRNs, which must now create tunnels with all associated PEs. We have accounted for the TRNs' capacity for cryptographic operations in our problem formulation. Furthermore, HIP uses ESP tunnels for the data plane to guarantee data confidentiality, source authentication, and integrity across the network. Additionally, since CEs are under a single ownership domain, implementing encryption at the CE side can enhance data protection without relying solely on the network provider.

B. TRN Relevance to IIoT Environments

- *Wide-Area, Large-Scale IIoT Deployments*: IIoT environments often comprise geographically dispersed devices requiring near-real-time control and monitoring. TRN architecture mitigates the "N-square" full-mesh scalability issue by significantly reducing the number of required tunnels and reducing the expansion of forwarding tables on PEs. This optimization translates to more efficient resource utilization and lower operating expenses when scaling out IIoT infrastructure.
- *IIoT Mission-Critical Applications*: Industrial processes are often sensitive to attacks and interruptions, necessitating robust encryption and authentication. By incorporating HIP at the foundation of our TRN-based framework, the proposed system maintains end-to-end data integrity and confidentiality while also mitigating DDoS threats through HIP's cryptographic namespace. Moreover, the concentration of relay functionality in TRNs minimizes potential attack surfaces and simplifies security management across large deployments.
- *IIoT Low Latency and High Reliability Requirements*: Many IIoT applications demand near-real-time data exchange and must continue operating without interruption in the event of equipment failures. By locating TRNs and using efficient tunnel scheduling, the additional latency (or path stretch) introduced is minimized. Furthermore, incorporating redundancy into the TRN architecture supports rapid failover, enabling ongoing, reliable, and secure communication even under network stress.
- *Resource-Constrained Endpoints with High Data Volumes*: IIoT edge devices commonly rely on an intermediate edge layer for data encryption, aggregation, and processing, as these devices often have limited computational resources. While the gateway typically handles real-time analytics and local device management tasks, it is advantageous to minimize its load related to network connectivity overhead. In our proposed design, most of the encryption and routing responsibilities are transferred to the TRNs, thereby reducing the number of tunnels and forwarding states that each gateway must manage. By offloading these tasks to the TRN, the gateway can dedicate its available processing capacity primarily to device-related functions, ultimately supporting a larger user base and facilitating secure, large-scale IIoT deployments.

C. Other Applications

Our formulation applications extend beyond the HIPLS addressed in this paper. It may be used in other areas. For the sake of brevity, we list some of them as follows:

- *Cloud VPN Gateways:* The VPN gateway provides network connectivity services to establish a secure and reliable connection from customer sites (e.g., data centers and office networks) to the Virtual Private Cloud (VPC). IPsec-VPN establishes a secure tunnel for data traversing between VPCs and customer sites. In dual-tunnel mode, multiple on-premises gateways are present at the customer's site to establish encrypted tunnels to the VPCs. These gateways can function in either active or standby mode. Our formulation, with minor modifications, can be applied to reduce the costs associated with activating and deactivating gateways. Additionally, it facilitates optimal allocation of customer sites to gateways, ensuring secure connectivity to the cloud. This approach paves the way for minimizing the expenses incurred in cloud services under pay-by-data-transfer agreements, thereby reducing overall costs.
- *Private Relay Services:* Private relay employs encryption and recent transport enhancements to relay user device traffic through a combination of proprietary and partner infrastructures before directing this traffic to the intended website. Utilizing a chain of relays, this multi-hop relay architecture is designed to protect end-user privacy. Similarly, our proposed formulation can be used to determine the active relays within a specific region and the allocation of users to the assigned relay nodes.
- *Industrial IoT Gateway Selection*: Gateways in IoT networks serve as intermediaries between IoT devices and cloud providers, transferring traffic and managing preprocessing. Efficiently managing the data flow from heterogeneous IoT devices to cloud providers relies on optimizing gateway selections. Our proposed method can efficiently identify the most cost-effective primary gateways by considering activation/deactivation costs, association costs between IoT devices and gateways, network conditions, and device distribution.

D. K-tunnel Relay Placement Problem

1) General median K-tunnel relay placement problem: This section explains the general problem of locating the median tunnel relays in the provider network graph. Then, with a minor modification, we will update the K-tunnel relay placement problem to include failure resistance.

Network model: The provider network is modelled as an undirected graph $G = (V, E)$, where V denotes the set of routers (both PE and P routers) and E the set of links. Each edge $e \in E$ is associated with a positive weight $w_e > 0$ (e.g., latency or monetary cost). Let $\mathcal{J} \subset V$ be the set of PE routers.

Shortest–path metric: For any $u, v \in V$ we write $d(u, v)$ for the length of the minimum-weight u–v path with respect to the weights w_e.

Placement definition: A placement of K TRNs is a set $R_K = \{r_1, \ldots, r_K\}$ with $|R_K| = K$, where each r_i may lie anywhere on the edge of G or coincide with a vertex. For any PE router $j \in \mathcal{J}$ and placement R_K, we define

$$D(R_K, j) = \min_{1 \le i \le K} d(r_i, j)$$

to be the distance from PE j to its nearest relay in R_K. The total distance of a placement is

$$\mathcal{D}(R_K) = \sum_{j \in \mathcal{J}} D(R_K, j)$$

Optimization goal: The median K-TRN placement problem seeks a placement R_K^* minimizing this sum, i.e.,

$$\mathcal{D}(R_K^*) = \min_{R_K \subseteq |G|,\ |R_K| = K} \mathcal{D}(R_K)$$

where $|G|$ is the geometric realization of the provider network and includes all points on the vertices and edges of G.

Vertex-optimality: Although R_K is allowed to contain points on edges (the absolute median formulation), Hakimi [27] proved that an optimal solution always exists with all relay points located on vertices. Hence, we may restrict attention to vertex placements:

$$\min_{R_K \subseteq |G|} \mathcal{D}(R_K) \;=\; \min_{\substack{M_K \subseteq V \\ |M_K| = K}} \mathcal{D}(M_K)$$

where M_K denotes a K-subset of vertices. Placing relays on edges would require physical infrastructure changes and higher CAPEX/OPEX; therefore, in what follows, we study the problem with relays confined to vertices only.

2) Complexity of the median problem on the provider network: The general median K-tunnel relay placement problem on a general network is well known to be NP-hard when K is part of the input and can grow with $|V|$. However, for the special case where the network is a tree, polynomial-time algorithms exist for arbitrary K. On the other hand, if K is fixed as a small constant (and does not grow with the input size), then one can solve the K-median problem in time polynomial in $|V|$, for instance by enumerating $\binom{|V|}{K}$ [28], [29].

3) Reliable K-Tunnel Relay placement:

Table II
SUMMARY OF MAIN NOTATIONS

Notation	Description
a_i	Activation cost of TRN i
c_{ij}	Cost of serving one unit of demand for PE j by TRN i
$d(r_i, j)$	Shortest path between PE j and TRN r_i
d_j	Demand of PE j
$D(R_K, j)$	Distance between PE j to one designated relay in R_K
$D_{R\text{-}TRP}(R_K, j)$	Distance between PE j to all R_K in R-TRP
$\mathbf{D}$	Minimum distance matrix of G for PEs
$\mathcal{D}(R_K)$	Sum of distances from all PEs to their associated TRNs in R_K
$\mathcal{D}_{R\text{-}TRP}(R_K)$	Distance-sum from all PEs to all R_K in R-TRP
$\mathcal{I}$	Optimal location of TRNs (Algorithm 1)
$\mathcal{J}$	Set of PE routers in the network
μ	Lagrangian multiplier
$G = (V, E)$	Undirected graph where V is the set of PE/P routers and E is the set of links
$g(j)$	Weight attached to each PE j
I_A	Set of active TRNs
I_D	Set of inactive TRNs
M_K	Set of TRNs that exists only on vertices of G
o_i	Deactivation cost of TRN i
R_K	Set of a K TRNs (placement)
R_K^*	Optimal placement set of a K TRNs
s_i	Capacity of TRN i
w_e	Cost (latency) of the links
x_{ij}	Fraction of PE j demands served by TRN i
y_i	Variable indicating TRN i is active
z_i	Variable indicating TRN i is inactive

Motivation: In HIPLS, each PE router needs to support many CEs (up to hundreds/thousands). This yields the establishment of a secure HIP tunnel between the participating PEs that connects the geographically distant CE sites. In the event of a failure in a TRN, without backup plans in place, communication breakdowns might occur, resulting in significant costs for customers and providers. To ensure continuity, we extend the basic median formulation so that every PE has *multiple* relays in reach.

Problem modification: To address the limitations of this approach and meet the requirements of fault-tolerance relay-based communication in HIPLS, we propose a modified version of the median problem named Reliable K-Tunnel Relay Placement (R-TRP). This modified approach involves connecting each PE to all TRNs, thereby establishing an all-connectivity requirement that enables and enhances failure resilience in the event of a breakdown of active TRNs. We note that the optimization of selecting active and inactive TRNs and scheduling TRNs will be considered in separate problem formulations outlined in Section III-E. The main notations used in the paper are listed in Table II.

Formal definition: R-TRP considers only PE routers as clients; P routers act purely as transit nodes. Let $g(j)$ be the number of CEs connecting to PE $j \in \mathcal{J}$, and let K be the desired number of TRNs supplied by the operator. Given a placement $R_K = \{r_1, \ldots, r_K\}$, we define the distance

$$D_{\text{R-TRP}}(R_K, j) \;=\; \sum_{i=1}^{K} d(r_i, j)$$

Algorithm 1 Reliable K-Tunnel Relay placement

Require: $G(V, E)$, w_e, $g(j)$, K
Ensure: Location of K TRNs on vertices of G
1: Calculate minimum distance matrix $\mathbf{D}$ of G for PEs
2: Obtain matrix $\mathbf{D}'$ by multiplying each column of matrix $\mathbf{D}$ by PE weight $g(j)$
3: Compute the row-wise sum for each row in $\mathbf{D}'$
4: Sort rows of $\mathbf{D}'$ based on their row-wise sums in ascending order
5: $\mathcal{I}$= vertices corresponding to the first K rows in the sorted $\mathbf{D}'$
6: **return** $\mathcal{I}$ ▷ Return the placement of K TRNs

as the sum of shortest-path lengths from PE j to *all* relays. The weighted network-wide cost is

$$\mathcal{D}_{\text{R-TRP}}(R_K) = \sum_{j \in \mathcal{J}} g(j)\, D_{\text{R-TRP}}(R_K, j)$$

The optimization goal is to find a placement R_K^* that minimizes the cost of:

$$\mathcal{D}_{\text{R-TRP}}(R_K^*) = \min_{R_K \subseteq V,\, |R_K|=K} \mathcal{D}_{\text{R-TRP}}(R_K)$$

In other words, R-TRP problem is to find a placement of K TRNs such that each PE should connect to all K TRNs and the sum of connection cost with respect to link cost w_e between all PEs and TRNs is minimized.

Reliability rationale: We need to mention that our requirement for each PE to connect to every TRN arises from reliability considerations. Specifically, in this step, we place K TRNs such that each PE has both primary and backup relay nodes available if failures occur. In practice, however, not all TRNs must remain active or carry traffic for each PE at all times. As we will show later in Section III-E, we impose capacity constraints and manage activation/deactivation costs for TRNs. Consequently, while each PE has multiple TRNs available, each PE may be served by only one (or a subset) of these TRNs in any given scheduling period. If only one TRN per PE were allowed at this stage, no backup connectivity would be possible, and in some cases, no feasible solution could exist under failure scenarios.

In Algorithm 1, we calculate the minimum distance matrix for G. Each row of this matrix represents a candidate placement (vertices), while each column corresponds to a PE. We then adjust the matrix by multiplying each column by the number of customer sites supported by each PE, thus giving more weight to PEs supporting more customer sites. Finally, the first K rows, sorted by the minimum summed distance, display the candidate location for TRNs with the smallest total distance to all PEs, as detailed in lines 3-5. Algorithm 1 achieves a time complexity of $O(V^3)$, primarily attributed to the computation of the all-pairs minimum distance matrix (e.g., using Floyd-Warshall algorithm) in Line 1. Moreover, in Algorithm 1, K is treated as a given parameter (e.g., specified by the network operator based on budget or reliability demands).

E. Optimization of TRNs

We consider a set of K TRNs controlled by the network operator, thus knowing the network parameters. After placement (Algorithm 1 in Section III-D3), the demands of the PEs must be satisfied by the K TRNs. The required bandwidth for data transfer over encrypted tunnel to other PEs is amongst examples of PE's demands. Upon executing Algorithm 1, the optimal location of TRNs is denoted by $\mathcal{I}$.

The network operator decides on the allocations of these PE's demands to TRNs, i.e., acts as a schedule of the tunnel resources. Each TRN $i \in \mathcal{I} = \{1, \cdots, K\}$ can serve at most s_i resources and be active or inactive (idle) within the scheduling period. The PE $j \in \mathcal{J} = \{1, \cdots, J\}$ demands an integer amount d_j of tunnels (bandwidth) for its operation. Decision variables x_{ij} model allocations, i.e., the fraction of the demand of j-th PE served by the i-th TRN. The cost of serving one unit of demand for PE j by TRN i is denoted by c_{ij}. The activation cost of the i-th TRN is a_i. When it is active, the binary decision variable y_i is set to one; otherwise, it is set to zero. Similarly, the deactivation cost of the i-th TRN is o_i.

When the i-th TRN is deactivated, the binary decision variable z_i is set to one; otherwise, it is set to zero. A network operator may calculate the activation and deactivation costs based on factors such as the number of HIP-IPsec tunnels requiring creation, maintenance, or forcible closure. Moreover, in practice, if the tunnel scheduling problem is re-run each time network demands change, an operator can incorporate any previously active or inactive TRNs by updating the parameters a_i and o_i accordingly. For instance, if a TRN i remains active from a prior scheduling run, its activation cost a_i can be set to zero, since no additional overhead is needed. Similarly, if a TRN remains inactive, one may set its deactivation cost o_i to zero in the current run.

The following optimization amounts to minimizing the cost of activating the TRNs, the cost of serving PE's demands, and the cost of terminations of TRNs based on the network status.

$$(TS)\ C_{TS}^* = \min_{y_i, x_{ij}, z_i} \sum_{i \in \mathcal{I}} a_i y_i + \sum_{i \in \mathcal{I}} \sum_{j \in \mathcal{J}} c_{ij} x_{ij} + \sum_{i \in \mathcal{I}} o_i z_i$$

$$\text{s.t.} \sum_{i \in \mathcal{I}} x_{ij} = 1,\ \forall j \in \mathcal{J}, \tag{1a}$$

$$\sum_{j \in \mathcal{J}} d_j x_{ij} \le s_i y_i,\ \forall i \in \mathcal{I}, \tag{1b}$$

$$y_i + z_i = 1,\ \forall i \in \mathcal{I}, \tag{1c}$$

$$0 \le x_{ij} \le 1,\ \forall i \in \mathcal{I},\ \forall j \in \mathcal{J}, \tag{1d}$$

$$y_i \in \{0, 1\},\ \forall i \in \mathcal{I}, \tag{1e}$$

$$z_i \in \{0, 1\},\ \forall i \in \mathcal{I}. \tag{1f}$$

Constraint (1a) guarantees that every PE connects to a primary TRN. Constraint (1b) ensures that the TRNs have sufficient capacity (e.g., bandwidth, number of supported tunnels) to fulfill all the demands of PEs in the network. In other words, each TRN is unable to meet the demands of the PEs if these demands exceed its capacity. In real deployments, s_i typically denotes the number of simultaneous HIP-IPsec tunnels (or total bandwidth) each TRN can handle. Due to this limit, a

single TRN generally cannot serve the entire network alone, necessitating multiple TRNs to share the load. However, depending on the traffic demands at any given time, a subset of the TRNs may suffice to carry the load, and the remaining can be deactivated to reduce costs. If $\sum_i s_i < \sum_j d_j$, then (TS) is infeasible, and the provider should allocate additional resources to the TRNs to fulfill all PEs' demands.

Constraint (1c) enforces the relationship between z_i and y_i decision variables so that a TRN would be active or inactive. The last three constraints delineate the domains of decision variables: the allocations of PEs to TRNs, the activations, and the deactivations of TRNs.

Proposition III.1. *Tunnel Scheduling (TS) Problem is NP-complete.*

Proof. (a) Verifying the feasibility of a given solution to the problem (TS) and whether its associated objective function value is at most p can be efficiently determined in polynomial time relative to the problem size. Hence, (TS) is in NP.
(b) Any instance of the Capacitated Facility Location Problem (CFLP) can be polynomially reduced to the problem (TS). This means that an instance of CFLP can be solved as an instance of problem (TS). Given a CFLP instance:

$$\begin{aligned}(CFLP)\ \min_{x_{ij},w_i} & \sum_{i\in\mathcal{F}}\sum_{j\in\mathcal{C}} c_{ij}x_{ij} + \sum_{i\in\mathcal{F}} f_i w_i \\ \text{s.t.} & \sum_{i\in\mathcal{F}} x_{ij} = q_j, \forall j \in \mathcal{C}, \\ & \sum_{j\in\mathcal{C}} x_{ij} \le u_i w_i, \forall i \in \mathcal{F}, \\ & x_{ij} \ge 0, \forall i \in \mathcal{F}, \forall j \in \mathcal{C}, \\ & w_i \in \{0,1\}, \forall i \in \mathcal{F}.\end{aligned}$$

where $\mathcal{F} = \{1, ..., m\}$ is a set of potential facilities and $\mathcal{C} = \{1, ..., n\}$ is a set of clients. By setting: $\mathcal{I} = \mathcal{F},\ \mathcal{J} = \mathcal{C}, (d_j = q_j,\ \forall j \in \mathcal{J}), (w_i = y_i, a_i = f_i, s_i = u_i, o_i = 0,\ \forall i \in \mathcal{I})$, and rearranging the first two constraints of the CFLP instance, we obtain an instance of (TS). The solution (x^*, y^*, z^*) that is optimal for the (TS) instance also solves the CFLP optimally. Therefore, CFLP can be reduced polynomially to (TS). From (a) and (b), we get the conclusion. ■

IV. Algorithms for Solving (TS)

A. Optimal Solution

Since (TS) is NP-complete, no efficient approach is available to solve (TS) optimally for all input sizes unless $P = NP$. We obtain the exact solution for small size (TS) using the Gurobi solver. The optimal solution of (TS) can be found by an exhaustive search for all possible TRN activation/deactivation. Gurobi uses exact algorithms to attack (TS). The exact algorithm is guaranteed to find the optimal solution, but it can take a significant amount of time to provide it, which is not acceptable in the provider network.

B. Lagrangian-based Algorithm

Lagrangian relaxation exploits the structure of an integer programming problem by decomposing it into easier-to-solve subproblems. It identifies a subset of "hard" constraints that complicate an otherwise easy-to-solve problem. These constraints are relaxed by incorporating them into the objective function with Lagrange multipliers, penalizing their violation. The resulting Lagrangian problem is easier to solve and yields a lower bound (for minimization problems) on the optimal solution of the original problem [30].

In designing a Lagrangian relaxation algorithm, there are three key questions: (i) which constraints we need to select to be relaxed, as discussed in Section B1; (ii) how to derive a feasible solution for the original problem, given a solution to the relaxed problem, as outlined in Section B2; and (iii) how to update the Lagrangian multipliers to improve the solution quality, as detailed in Section B3.

To facilitate the decomposition of (TS) into smaller, easier-to-solve problems, we propose an algorithm that utilizes Lagrangian relaxation and subgradient optimization. This approach contrasts with branch-and-bound methods, which tend to be computationally expensive and harder to implement. After decomposing, we treat subproblems as standalone, independent problems and use well-established algorithms to solve them.

1) Lagrangian Relaxation: Constraint (1b) in (TS) is a candidate for relaxation. This decision is motivated by the fact that the constraint (1b) is a hard constraint. Furthermore, it has been shown that for problems similar to (TS) but without constraint (1c), the relaxation of constraint (1b) leads to a stronger tightness of the bound when considering the possible combinations of constraints to be relaxed [31]. This, in turn, provides better solutions to the original problem from theoretical and computational perspectives. Hence, by relaxing constraint (1b) and denoting the Lagrangian multiplier by μ, we introduce as problem $DC(\mu)$ the following Lagrangian relaxation of (TS):

$$\begin{aligned}C^*_{DC(\mu)} = \min_{y_i,x_{ij},z_i} & \sum_{i\in\mathcal{I}} \Big[(a_i - b_i)y_i + \sum_{j\in\mathcal{J}} (c_{ij} + q_{ij})x_{ij} + o_i z_i\Big] \\ \text{s.t.} & \sum_{i\in\mathcal{I}} x_{ij} = 1,\ \forall j \in \mathcal{J}, && \text{(2a)} \\ & y_i + z_i = 1,\ \forall i \in \mathcal{I}, && \text{(2b)} \\ & 0 \le x_{ij} \le 1,\ \forall i \in \mathcal{I},\ \forall j \in \mathcal{J}, && \text{(2c)} \\ & y_i \in \{0,1\},\ \forall i \in \mathcal{I}, && \text{(2d)} \\ & z_i \in \{0,1\},\ \forall i \in \mathcal{I}. && \text{(2e)}\end{aligned}$$

where μ_i are entries for row-vector μ, $q_{ij} = \mu_i d_j$ and $b_i = \mu_i s_i$.

The optimality condition for $DC(\mu)$ yields $(y_i = 0, z_i = 1)$ if $(a_i - b_i \ge o_i)$ and $(y_i = 1, z_i = 0)$ if $(o_i > a_i - b_i)$. Hence, the Lagrangian relaxation $DC(\mu)$ can be decomposed into a $\mathcal{J}$ independent problems, for each PE as

$$\begin{aligned}(DC_j)\ \ C^*_{DC_j} = \min_{x_{ij}} & \sum_{i\in\mathcal{I}} (q_{ij} + c_{ij})x_{ij} \\ \text{s.t.} & \sum_{i\in\mathcal{I}} x_{ij} = 1, && \text{(3a)} \\ & 0 \le x_{ij} \le 1,\ \forall i \in \mathcal{I}. && \text{(3b)}\end{aligned}$$

Moreover, I_A denotes the active TRNs, for which $(y_i = 1, z_i = 0)$. Furthermore, I_D represents the deactivated TRNs, having $(y_i = 0, z_i = 1)$. Each (DC_j) is a linear programming problem. Finally, the optimal value of $DC(\mu)$ is

$$C^*_{DC(\mu)} = \sum_{i \in \mathcal{I}_\mathcal{A}} (a_i - b_i) + \sum_{i \in \mathcal{I}_\mathcal{D}} o_i + \sum_{j \in \mathcal{J}} C^*_{DC_j} \quad (4)$$

Algorithm 2 Feasible Solution Generation: FSG

Require: $\mathcal{I}_\mathcal{A}, \mathcal{I}_\mathcal{D}$

Ensure: Feasible solution for (TS)

Step I:

1: Using active TRNs set $\mathcal{I}_\mathcal{A}$ and $N >> 0$ solve:

$$(FP) \quad C^*_{FP} = \min_{x_{ij}, v_i} \sum_{i \in \mathcal{I}_\mathcal{A}} \sum_{j \in \mathcal{J}} c_{ij} x_{ij} + \sum_{i \in \mathcal{I}_\mathcal{A}} a_i + N \sum_{i \in \mathcal{I}_\mathcal{A}} v_i$$

$$\text{s.t.} \sum_{i \in \mathcal{I}_\mathcal{A}} x_{ij} = 1, \ \forall j \in \mathcal{J}, \quad (5a)$$

$$\sum_{j \in \mathcal{J}} d_j x_{ij} \leq s_i + v_i, \ \forall i \in \mathcal{I}_\mathcal{A}, \quad (5b)$$

$$v_i \geq 0, \ \forall i \in \mathcal{I}_\mathcal{A}, \quad (5c)$$

$$0 \leq x_{ij} \leq 1, \ \forall i \in \mathcal{I}_\mathcal{A}, \ \forall j \in \mathcal{J}. \quad (5d)$$

Step II:

2: **if** $v_i = 0$ for all i **then** ▷ Feasible Solution

$$C^*_{FP} = \sum_{i \in \mathcal{I}_\mathcal{A}} a_i + \sum_{i \in \mathcal{I}_\mathcal{A}} \sum_{j \in \mathcal{J}} c_{ij} x_{ij} + \sum_{i \in \mathcal{I}_\mathcal{D}} o_i$$

3: **return** C^*_{FP} ▷ feasible solution for (TS)

4: **else if** there is a $v_i > 0$ **then**

5: **GoTo Step III** ▷ Infeasible Solution

6: **end if**

Step III:

7: $\bar{i} = argmin\{(a_i - b_i)\}$ where $i, \bar{i} \in \mathcal{I}_\mathcal{D}$

8: Update set of active TRNs as $\mathcal{I}_\mathcal{A} = \mathcal{I}_\mathcal{A} \cup \{\bar{i}\}$

9: **GoTo Step I**

2) Generating feasible solution for (TS): A solution to $DC(\mu)$ in Section IV-B1 does not necessarily provide a feasible solution to the primal problem (TS). Therefore, we propose the subsequent feasibility problem (FP) that yields a feasible solution for (TS). The latter method employs auxiliary variables v_i to determine whether the solution of $DC(\mu)$ violates any capacity of the TRNs. If a capacity violation occurs, an inactive TRN is reactivated to mitigate it. This process is repeated until a feasible solution for (TS) is generated that entirely eradicates capacity violations.

Algorithm 2 represents steps to generate a feasible solution for (TS). By solving linear programming problem (FP) for activated TRNs (Line 1) and examining the corresponding v_i, we deduce if there is a potential violation of the capacity constraints of the TRN as follows: *Case 1):* The absence of $v_i > 0$ in the solution for (FP) signifies that the capacity constraints of the tunnel relay nodes are not violated; therefore, the solution is feasible for the primal (TS) problem, which is expressed in Line 2. *Case 2):* A capacity violation occurs when a value of $v_i > 0$ is present in the solution of (FP) (Line 4), rendering the infeasibility of the solution for the (TS) problem. When faced with infeasibility, Algorithm 2 selects an inactive TRN with the minimum coefficient $(a_i - b_i)$ and activates it (Line 7). This activated node is then added to the list of active TRNs $\mathcal{I}_\mathcal{A}$. This procedure increases the constraint capacity and allows the algorithm to restart from Step I, ensuring a feasible solution to the primal problem (TS) is obtained.

3) Updating Lagrangian multiplier: Lagrangian relaxation, which is a relaxation for the primal problem (TS), provides the lower bound for the optimal solution of (TS), denoted as $C^*_{DC(\mu)} \leq C^*_{TS}$. Furthermore, feasible solution obtained in Section IV-B2 provides an upper bound for the optimal solution of (TS), it follows that $C^*_{FP} \geq C^*_{TS} \geq C^*_{DC(\mu)}$. Improving the value of $C^*_{DC(\mu)}$ leads to an improvement in the quality of the solution obtained for (TS). To achieve this objective, we define the Lagrangian dual as follows.

$$(DP) \quad C^*_{DP} = \max_{\mu \geq 0} \ C^*_{DC(\mu)} \quad (6)$$

The dual program (DP) is piecewise linear and therefore not differentiable in the entire domain [32], [33]. We employ subgradient optimization to find the optimal value of the Lagrangian multiplier μ that maximizes the objective function of (DP). In other words, we calculate a series of $\mu^k (k \in \mathbb{N})$ in which (DP) converges to the optimal solution by calculating the subgradient [34] using

$$\beta_i = \sum_{j \in \mathcal{J}} d_j x^k_{ij} - s_i y^k_i \quad (7)$$

and the step size as

$$t_k = \frac{\lambda(\mathcal{UB} - DC(\mu^k))}{\sum_{i \in \mathcal{I}} ||\beta_i||^2} \quad (8)$$

where x^k_{ij} and y^k_i represents optimal solution to $DC(\mu^k)$. Furthermore, $\mathcal{UB}$ denotes the best upper bound found up to iteration k. μ^k represents the multiplier value in iteration k, respectively, and we have $0 < \lambda \leq 2$. Finally, subgradient optimization updates the Lagrangian multiplier for the next iteration as follows:

$$\mu^{k+1}_i = max\{0, \mu^k_i + t_k \beta_i\} \quad (9)$$

4) Computational Complexity and Scalability of the LA-TS: The proposed algorithm to solve (TS) is described in Algorithm 3. The proposed Lagrangian-based algorithm (LA-TS) is an iterative algorithm that terminates with an approximate solution to (TS) when either the chosen gap or the chosen number of iterations is met (see Algorithm 3). For each iteration: LA-TS solves $|\mathcal{J}|$ independent linear problems (LP) at step 1 and at most $|\mathcal{I}|$ LP problems at step 3. Consequently, the computational complexity of the proposed algorithm is polynomial, and hence, it is scalable in the size of the configuration in terms of the CPU requirements. Figure 3 depicts a concise roadmap of how the three algorithms interlink to deliver our scalable and secure VPLS architecture.

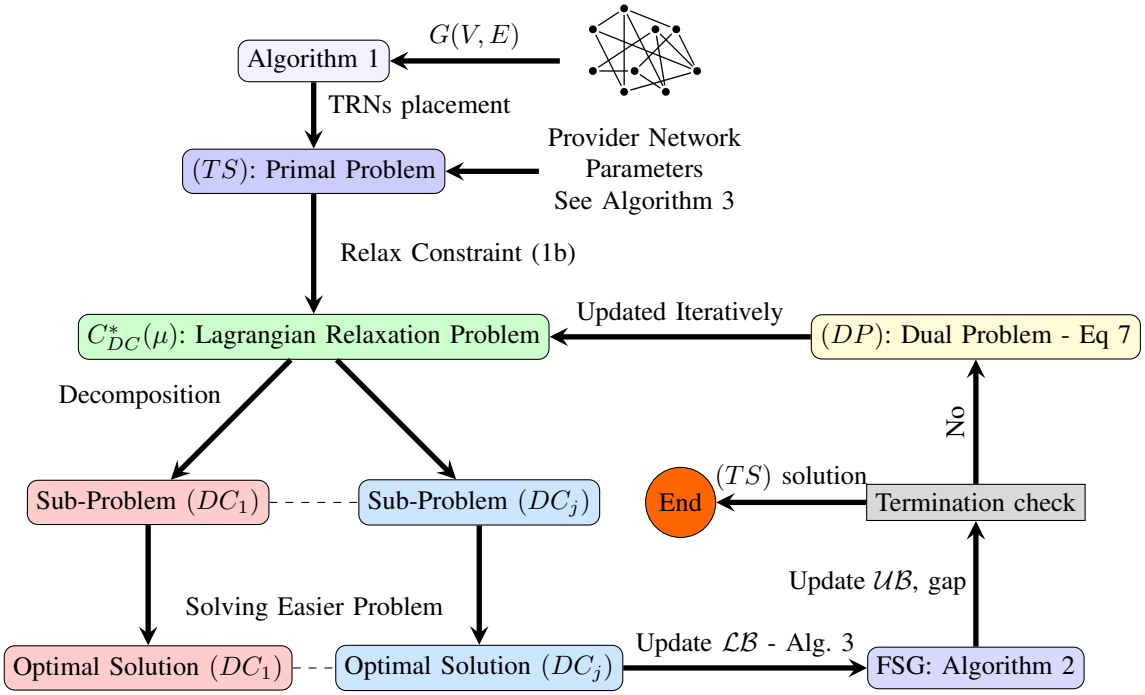

Figure 3. Comprehensive workflow of the proposed solution, highlighting process flow in highlighting TRN selection (Algorithm 1), feasible solution generation (Algorithm 2), and scheduling (Algorithm 3).

Algorithm 3 Lagrangian-based Algorithm: LA-TS

Require: a_i, c_{ij}, o_i, s_i. Initialize $k = 0$, μ^k , $\mathcal{UB}$, $\mathcal{LB}$, stopping conditions *Iters* and ϵ

Ensure: TRNs (Act/Deact)ivation, PEs association to TRNs

1: Solve $DC(\mu^k)$ - Section IV-B1.
2: Update $\mathcal{LB} = C^*_{DC(\mu^k)}$ if $\mathcal{LB} < C^*_{DC(\mu^k)}$.
3: Generate feasible solution C^*_{FP} to (TS) - Algorithm 2.
4: gap= $\frac{\mathcal{UB}-\mathcal{LB}}{\mathcal{LB}}$. Update $\mathcal{UB} = C^*_{FP}$ if $\mathcal{UB} > C^*_{FP}$.
5: Terminate if gap$\leq \epsilon$ or $k \geq Iters$.
6: Update the Lagrangian multipliers using $(7) - (9)$.
7: Update $k = k + 1$.
8: *Goto 1*.
9: **return** $\mathcal{UB}$ ▷ Solution for (TS)

V. Evaluation and Discussion

A. Evaluation Setup

To evaluate the performance of our proposed algorithms, we used the Gurobi exact solver and implemented our proposed algorithm in Python. Gurobi was used to evaluate the algorithm's running time and efficiency and to compare the objective value of the (TS) problem. Specifically, we compared the cost of our solution to the optimal cost (objective value function calculated by Gurobi). The experiments were conducted using a PC operating on Windows 10, equipped with an AMD Ryzen 7 PRO processor and 32 GB of RAM. In the evaluation section, we utilize the following network topologies and configurations:

- *Network topology*: Given that secure VPLS (for IIoT deployments) is mainly used on large-scale networks, it becomes crucial for our evaluation to incorporate real network topologies. Therefore, we have chosen to utilize the Internet Topology Zoo [35], an ongoing project to gather data on network topologies from various global locations. The real network topologies selected for this study include Cogent (spanning the USA and Europe), Bell Canada (within Canada), and Colt (across Europe). In each of the network topologies, vertices represent routers, while edges show the communication links between these routers. Furthermore, the latency between each pair of connected routers is calculated based on their distance.
- *Traffic demands*: In a secure VPLS network, each PE router has a specific traffic demand that must be transmitted to some (or all) other PE routers participating in a single VPLS instance. This demand can be quantified as the amount of data flow, measured in megabits, traversing from one PE router to another. To populate the traffic demands matrix between PE routers, we used sndlib [36], an open-source platform known for providing realistic network traffic. The demand matrices generated by sndlib are based on precise traffic measurements obtained from actual IP networks while ensuring privacy through proper anonymization techniques. Moreover, in practical industrial deployments, IIoT devices such as sensors, conveyor belts, and robotic stations are typically connected to CE equipment positioned behind the PE routers. Each CE is protected by its designated PE through HIP-secured tunnel to TRNs. For demonstrative purposes, we use Factory I/O to simulate a discrete automation environment that includes conveyors, stations, and sensors controlled by

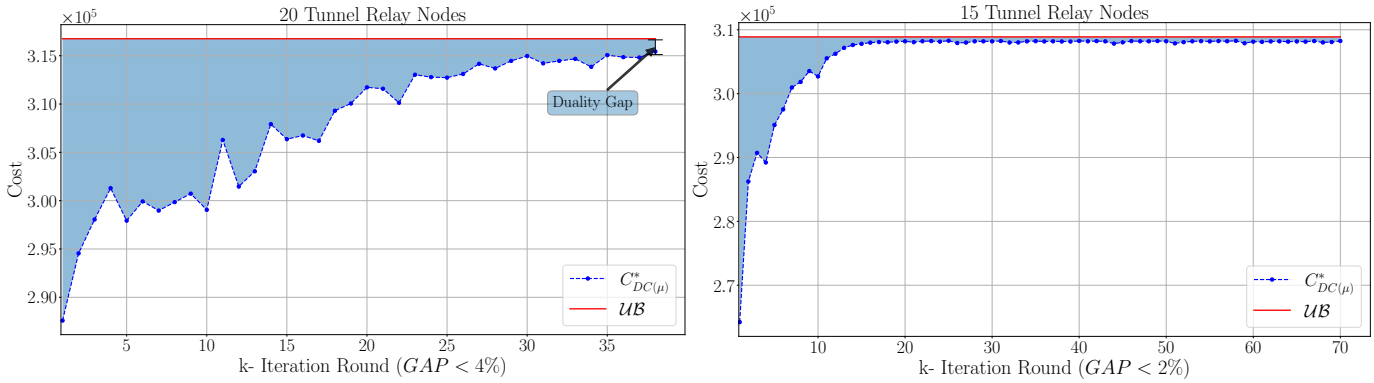

Figure 4. Convergence of $C^*_{DC(\mu)}$, alongside with the upper bound $\mathcal{UB}$ in Algorithm 3. The iterations are shown up to the point where the GAP attains a threshold of $< 4\%$ and $< 2\%$. The duality gap is $\mathcal{UB} - C^*_{DC(\mu)}$ in the final iteration, where $\mathcal{UB}$ is determined as the best C^*_{FP} found in Line 4 of Algorithm 3.

OpenPLC servers. The OpenPLC instance links remotely to the CEs, thus creating communication requests that translate into the traffic demands modeled in our (TS) formulation.

- *Activation/Deactivation costs of TRN*: Given that TRNs are placed within the network provider, the costs associated with their activation and deactivation can be configured by network administrators. TRNs may exist both as hardware solutions and software-based cloud services. Consequently, factors such as the number of current IPsec tunnels that need to be terminated or established on each TRN and the costs associated with powering off and on these TRNs are among the potential considerations for activation and deactivation costs.

B. Results

Figure 4 depicts the convergence behavior of our proposed Lagrangian-based algorithm (Algorithm 3) towards the optimal solution of the (TS) problem. The optimal solution for (TS) exists within the range $[C^*_{DC(\mu)}, \mathcal{UB}]$. Therefore, a smaller range results in better solution quality. The x-axis illustrates the number of iteration rounds in Algorithm 3, while the y-axis shows the cost of the solution generated by the algorithm. This cost embodies the activating and deactivating costs of TRNs and the association cost between TRNs and PEs.

Regarding convergence patterns, both networks exhibit a decline in cost during the initial iteration rounds, showing fast improvement toward an optimal solution. However, as the iterations progress, the rate of improvement gradually decreases. The convergence also demonstrates a non-monotonic pattern due to the subgradient optimization method. Additionally, the duality gap decreases as the iteration rounds increase. It is important to highlight that the number of iteration rounds can be configured by network administrators, thus enabling a balance between the computational resources and time expended and the obtained quality of the solution.

In Figure 4, convergence occurs within fewer k-Iteration Rounds for the network with 20 TRNs. This observation could be due to the algorithm's preset stopping criteria being met more quickly, possibly because of a larger initial gap or bounds that are easier to estimate in the case of a larger problem set.

As previously discussed, relaying (by employing TRNs) plays a crucial role in addressing the N-square scalability issue in VPLS networks by diminishing the creation of a full-mesh connectivity between PE routers. However, this approach incurs its own costs, as the end-to-end traversal of data packets results in increased latency for communication between customer sites.

Following the placement of TRNs within the provider's network, each PE designates the associated activated TRNs as its default gateways for secure data transmission to other PEs. Consequently, the placements of TRNs and their association with PEs directly affect the data paths that packets must traverse. We introduce the term *'Path Stretch Ratio'* to describe the ratio of the distance traveled via TRNs to the distance in direct full-mesh connectivity between PEs. Utilizing Algorithm 1 for TRN placement and Algorithm 3 for selecting active TRNs and associating PEs with TRNs, we evaluate the path stretch ratio across three distinct network topologies of varying sizes. Table III presents these ratios along with a 95% confidence interval. The findings indicate that, in the worst case, the path stretch ratio approaches 1.51. Given that Algorithm 1 accounts for long-term network traffic demands in a median context and considering the hub-spoke architecture dominant in VPN networks, our proposed methodology effectively limits the path stretch across all simulated networks. However, it is important to note that this increased end-to-end latency represents a trade-off that network providers and customers must consider in the large-scale deployment of secure VPLS. Moreover, the (TS) formulation itself can be re-weighted by adjusting c_{ij} and a_i to prioritize shorter tunnel paths when required, thus offering flexibility in balancing tunnel count against path length.

Figure 5 illustrates a comparison of solution cost and execution time for the Gurobi exact solver, which solves the joint placement and scheduling problem, and our proposed Lagrangian-based method (Algorithms 1 and 3) under increasing network sizes. The exact solver achieves the optimal cost for all tested configurations but exhibits a significant rise in

Table III
ALGORITHM 1-3 PATH STRETCH RATIO WITH 95% CONFIDENCE INTERVAL

Network	Location	#Nodes	Path Stretch	Error Margin
Cogent	USA-Europe	197	1.3736	0.1421
Bell Canada	Canada	48	1.5171	0.0839
Colt	Europe	153	1.293	0.0339

runtime when scaling to larger networks (reaching up to 257 seconds), which is impractical for dynamic and large-scale IIoT scenarios. In contrast, our Lagrangian-based solution retains an execution time on the order of one second even for large deployments and finishes in 0.02 seconds for networks with approximately 100 nodes. While this approach incurs a limited average cost gap of approximately 7% relative to the optimal solution, it offers substantial gains in speed and scalability, critical factors for rapid reconfiguration and real-time operations in large-scale IIoT environments.

Figure 5 supports the argument that the Lagrangian-based algorithm is more suitable for practical applications, especially when quick decision-making is vital, such as dynamic network routing, real-time resource allocation, or during emergency response when network configurations must be optimized promptly.

Figure 6 compares the solution costs obtained by Gurobi under a strict time limit against our Lagrangian-based approach (Algorithms 1 and 3). Specifically, we measure Gurobi's cost at the exact execution time taken by the Lagrangian method on each network size. To quantify the difference between the two, we define

$$\delta(\%) = \frac{\text{Cost}_{\text{Gurobi-Forced}} - \text{Cost}_{\text{Lagrangian}}}{\text{Cost}_{\text{Gurobi-Forced}}} \times 100\%,$$

where $\text{Cost}_{\text{Gurobi-Forced}}$ and $\text{Cost}_{\text{Lagrangian}}$ are the objective function values found by Gurobi and our Lagrangian approach, respectively.

From the figure, Gurobi can provide a feasible solution relatively fast under the same time budget. Nevertheless, the Lagrangian method typically achieves a noticeably lower cost, yielding δ values in the range of 14% to over 40% improvement. This indicates that while Gurobi's truncated run obtains a decent feasible solution, our approach consistently outperforms it in terms of objective value whenever both methods are capped by the same runtime.

Figure 7 presents a comparison of solution costs for a range of network sizes, evaluating our proposed solution (Algorithm 3) against the greedy approach (Algorithm 4). In the greedy approach, TRNs are inactive during initialization, and the solution cost is initialized to zero. Then, Algorithm 4 sorts TRNs based on the costs. For each PE, it searches for a TRN that can serve the PE's demand at the lowest cost. When a TRN is found, the algorithm activates it, serves the PE demand, and updates the solution cost. Finally, the algorithm repeats the TRN selection process iteratively until all PEs' demands are met or no additional TRN can be activated.

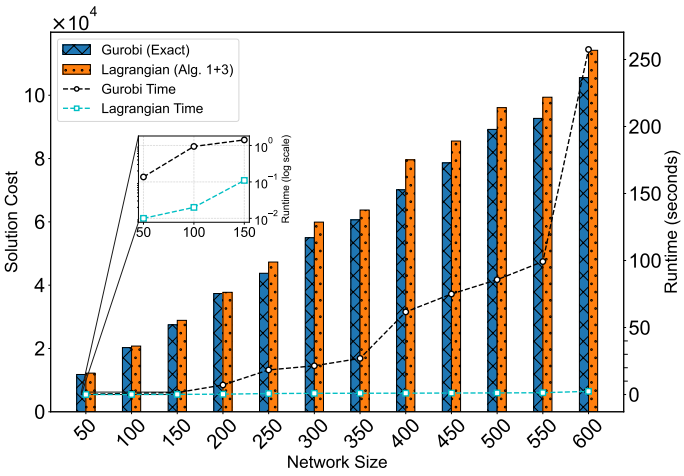

Figure 5. Comparison of solution cost (TS objective) and runtime for the Gurobi exact method versus our Lagrangian-based approach (Algorithms 1 and 3) as network size increases.

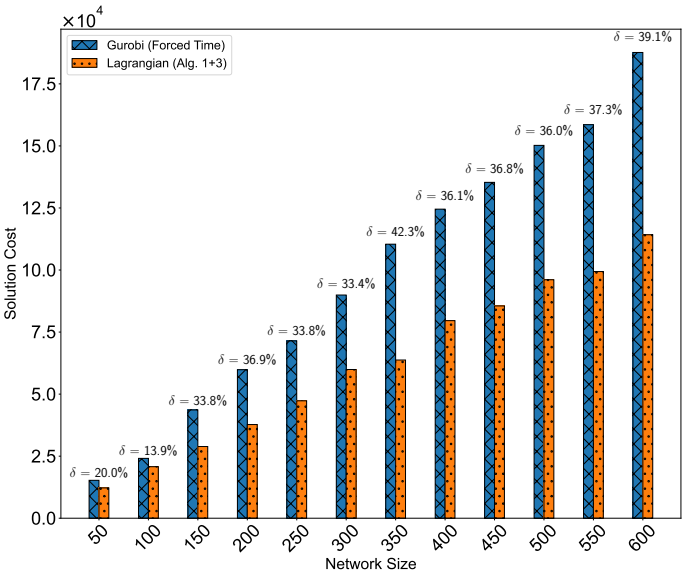

Figure 6. Comparison of solution costs under equal runtime for Gurobi (Forced Time) and the Lagrangian approach. δ denotes the percentage improvement of the Lagrangian method relative to Gurobi.

In Figure 7, the Lagrangian-based Algorithm 3 is depicted with two bounds: a lower bound LB indicated by green lines and an upper bound UB indicated by blue lines. These bounds encapsulate the potential solution space for the algorithm, with an optimal solution expected to exist within this range. It should be noted that the simulations employed variable random activations and deactivations, leading to a diversity of solution costs across the spectrum of network sizes.

The costs associated with the greedy approach are consistently higher than the UB of Algorithm 3, suggesting that while greedy Algorithm 4 may offer faster solutions, it does so at the expense of optimality. In practice, the optimal solution is expected between LB and UB. In the figure, we observe that the UB for Algorithm 3 is always below the cost found by Greedy Algorithm 4, underscoring the effectiveness of Algorithm 3 in providing cost-effective solutions. We need to

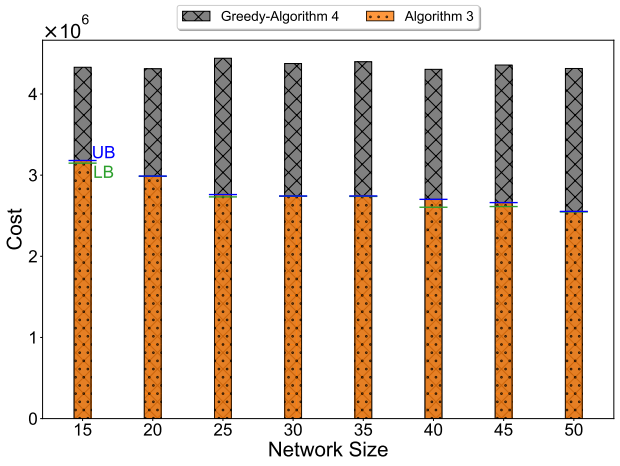

Figure 7. Solution cost comparison: Algorithm 3 vs. Greedy approach.

highlight that the gap between the LB and UB remains stable as the network size increases. This indicates that Algorithm 3 exhibits effective scalability with respect to the size of the problem.

Figure 8 presents a comparative analysis of the number of tunnels in our proposed approach against several existing solutions: the non-secure VPLS (BGP/LDP VPLS), secure VPLS draft based on the HIP (HIPLS) [16], its session-key enhanced variant (S-HIPLS) [37], and the Secure SDN-VPLS [38] which leverages Software-Defined Networking (SDN) to manage tunnel numbers. The lower part of Figure 8 illustrates the average number of tunnels per PE concerning the network size. In existing solutions, where full-mesh connectivity is required for each PE to all others, the number of tunnels per PE increases linearly with the addition of PEs to the network. The SDN-VPLS approach improves over this linear model, albeit the growth rate is only marginally reduced.

Our proposed solution, which seeks to place TRNs close to PEs and requires only TRNs to connect in a full-mesh with other PEs, demonstrates a distinct advantage over the existing solutions. This is evident by the substantially lower number of tunnels per PE across all network sizes, including small-scale deployments, thereby offering a more resource-efficient network utilization. This advantage highlights the potential of our proposed approach to reduce infrastructure demands and enhance scalability in secure VPLS environments. The upper part of Figure 8 shows the contrast between the total number of tunnels in the network when applying our proposed solution compared to the existing solutions that follow a quadratic growth pattern ($\frac{n\times(n-1)}{2}$) where n represents the number of PEs. BGP/LDP VPLS, HIPLS, and S-HIPLS show a quadratic increase in the number of tunnels as the number of PEs increases, which aligns with the expected full-mesh topology that requires a connection between every pair of nodes. Our proposed solution, which utilizes TRNs that only maintain full-mesh reachability and deactivate unnecessary TRNs, exhibits a significant departure from this direction. The result is a substantially flatter curve for Algorithm 1-3, even considering the worst-case scenario depicted. This

Algorithm 4 Greedy Algorithm

Require: a_i, c_{ij}, o_i, s_i

Ensure: (Act)Deactivation of TRN, PEs to Relays association

Step 1: Initialization

Set all $y_i = 0$ ▷ all TRNs closed

$cost = 0$ ▷ solution cost

Sort TRNs by $(a_i - o_i)$

Step 2: TRN Selection

For each PE $j \in \mathcal{J}$ **do**:

    Find the TRN i that has min $(a_i - b_i)$ and has enough capacity to serve d_j.

    If such a TRN is found, set $y_i = 1$

    Serve PE demand from TRN i and update $s_i = s_i - d_j$

    $cost = cost + a_i + c_{ij}x_{ij}$

end For

Step 3: Iteration

Repeat **Step 2** until all PE demands are satisfied or no more TRN can be opened.

For closed TRN $(y_i = 0)$ set $cost : cost + o_i$

return $cost$ ▷ solution cost for (TS)

indicates a considerable reduction in the required tunnels, directly implying a more efficient network configuration with potential cost savings on infrastructure and enhanced network maintenance due to reduced complexity. While deploying TRNs does introduce an additional baseline cost (e.g., as a software-based VPN gateway), our analysis indicates that this expense is significantly outweighed by the operational savings from eliminating numerous direct tunnels in a full-mesh design.

Table IV compares the total number of forwarding entries installed in the VPLS network for both full-mesh connectivity and our proposed method. Using the example from Figure 2 as a reference, the total number of forwarding entries in a full-mesh is calculated as the number of PEs multiplied by the forwarding entries installed in each PE. The latter term is calculated by the number of CEs in the network. For instance, Cognet and Bell Canada have installed nine and three Transit TRNs, respectively, while the Colt network has deployed six TRNs. In our proposed method, the total number of forwarding entries in the network is calculated by summing the forwarding entries installed for each individual PE (spoke) and the TRNs in the network. The forwarding entries for a PE consist of the number of its supported CEs plus the total number of TRNs, which serves as the default gateway. Additionally, since each TRN must include all routes to the CEs for customer sites, the number of forwarding entries in a TRN is the sum of all supported CEs in the network. More specifically, in the case of Cognet, which supports 197 PEs and 500 CEs, each TRN would need 500 forwarding entries to ensure complete connectivity. Assuming all TRNs are activated, the total number of forwarding entries installed on the network is calculated as follows: the sum of forwarding entries on PEs $(197 \times 9 + 500)$ and forwarding entries on TRNs (4500). It

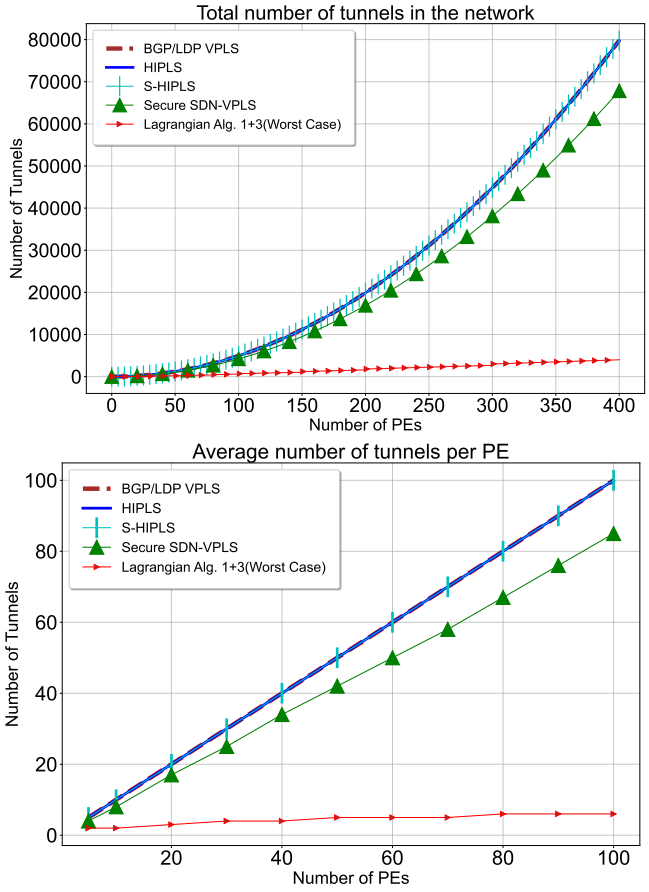

Figure 8. Number of required tunnels: our proposed approach vs. existing solutions.

is important to note that this analysis assumes a worst-case scenario in which all TRNs are activated during the scheduling period according to Algorithm 3. Compared to the number of forwarding entries installed in a full-mesh setup, our proposed approach, which incorporates relaying, achieves on average a 92% reduction in installed forwarding entries for all networks in Table IV. This significant decrease helps to conserve CAM memory, making it available for other network operations.

VI. Related Work

Recent advances in IIoT security have concentrated on securing device-level communications and establishing trustworthy gateways. For instance, Mahmood et al. and Zhang et al. proposed three-factor authentication and revocable fine-grained access control mechanisms to prevent unauthorized access and protect sensitive data in IIoT-enabled setups [39], [40]. Similarly, Fröhlich et al. and Zhang et al. integrate trusted execution environments and dynamic identity revocation to enhance the security of IIoT gateways against compromise [41], [42]. These approaches provide strong security measures at the application and transport layers; however, they do not adequately address the scalability issue at the network layer resulting from the expansion of the IIoT, especially in providing secure virtualized layer-2 overlay services such as VPLS.

Table IV
Comparison of total forwarding table entries: Our Proposed Method vs. Full-Mesh

Network	#PE	#CE	#entries full-mesh	#entries proposed method
Cogent	197	500	98500	6773
Bell Canada	48	100	4800	544
Colt	153	400	61200	3718

Other works explored network-oriented solutions, including SD-VPN overlays [43], in-network certificate validation using programmable data planes [44], and optimized VPN gateways equipped with anomaly detection [45]. While these approaches improve throughput and security monitoring, and centralization simplifies management and encryption, they can introduce a single bottleneck and do not inherently scale to extensive site-to-site IIoT communication. Our proposed solution utilizing TRNs aims to address secure communication of large-scale IIoT sites by jointly optimizing relay node placement and tunnel scheduling, allowing IIoT gateways to form secure, on-demand paths through minimal relay hops. This yields a more scalable and flexible site-to-site VPN fabric, as it reduces per-gateway tunnel load while preserving end-to-end encryption and authentication.

Yu et al. [46] proposed Edasvic, an efficient and dynamic storage verification scheme designed for cloud-based Industrial Internet platforms. By combining polynomial commitments with a lightweight homomorphic authenticator and an authenticator accumulator, Edasvic ensures data integrity while minimizing computational overhead on fog nodes and supporting dynamic data updates. While their work focuses on secure data verification in storage, our approach addresses a complementary yet distinct challenge: the secure and scalable transmission of IIoT data over VPLS. By optimizing tunnel relaying and scheduling, our work targets network-layer bottlenecks such as a high number of required tunnels and fast growth of the forwarding table, offering end-to-end scalability that supports reliable data delivery before it reaches the cloud.

Previous studies [24], [37], [47] have accurately pinpointed the scalability issue associated with full-mesh tunneling in HIPLS, proposing a hierarchical design as a potential solution to reduce the number of tunnels. However, these studies neither offered algorithmic solutions for determining the number of routers involved in creating the full-mesh network nor reported any mechanisms for resource allocation. Authors in [14], [48] considered relay node selection in VPNs as a constrained optimization problem, focusing on router uplink memory and bandwidth, as well as the additional distance traversed by data packets. These studies have demonstrated that relaying is generally a suitable method for VPNs due to the sparsity of their traffic demand matrix [49]. However, there remains a noticeable gap in research addressing the combined optimization of relay location selection and tunnel scheduling.

VII. Conclusions and Future Work

This paper presents a novel approach to addressing the inherent challenges in the scalability and complexity of secure

VPLS networks, particularly in large-scale IIoT deployments. In order to address the N-square scalability problem and the expansion of PE forwarding table entries, we have proposed the utilization of TRNs. Compared to existing solutions, our proposed approach demonstrates a significant reduction in the number of required tunnels and forwarding entries. Additionally, it simplifies the maintenance of secure VPLS, along with an improved balance between load distribution and failure resistance, which is crucial for facilitating efficient and secure communication in IIoT environments.

We demonstrated that, with minor modifications, the problem of placing TRNs in the network while ensuring failure resistance can be formulated as a modified median problem and solved in polynomial time. Furthermore, we proved that scheduling TRNs is an NP-complete problem. To address this complexity, we employed Lagrangian-based algorithms, which not only deliver high-quality solutions efficiently but also offer the flexibility to balance the trade-off between running speed and the quality of the generated solutions. The practical applicability of our approach is validated through extensive evaluation using real-world network topologies and traffic matrices. This highlights the feasibility and effectiveness of our proposed solutions in real-world scenarios, paving the way for their adoption in industrial settings.

While our proposed TRN-based solution significantly reduces tunnel and forwarding table complexities in large-scale deployments, certain limitations remain. First, the introduction of relay nodes increases end-to-end latency due to longer tunnel paths compared to full-mesh connection. Although we have demonstrated that this path stretch remains modest across real-world topologies, it may still impact latency-sensitive IIoT applications. Second, our current scheduling mode requires periodically updated network demands, which may not fully capture highly dynamic or bursty traffic patterns in some industrial environments. Finally, while security considerations have been addressed, future work could further investigate potential attack surfaces introduced by TRNs, including risks associated with compromised relays. Addressing these aspects will further enhance the robustness and applicability of HIPLS in diverse IIoT scenarios. Moreover, future studies might focus on dynamic relay node selection based on other real-time network conditions or investigate the integration of machine learning algorithms to predict and manage network demands more effectively.

REFERENCES

[1] H. Yu, R. Xu, H. Zhang, Z. Yang, and H. Liu, "EV-FL: Efficient Verifiable Federated Learning With Weighted Aggregation for Industrial IoT Networks," *IEEE/ACM Transactions on Networking*, pp. 1–15, 2023.

[2] M. N. Halgamuge, "Leveraging Deep Learning to Strengthen the Cyber-Resilience of Renewable Energy Supply Chains: A Survey," *IEEE Communications Surveys & Tutorials*, 2024.

[3] A. T. A. Ghazo and R. Kumar, "ANDVI: Automated Network Device and Vulnerability Identification in SCADA/ICS by Passive Monitoring," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2024.

[4] B. Babayigit and M. Abubaker, "Industrial Internet of Things: A Review of Improvements Over Traditional SCADA Systems for Industrial Automation," *IEEE Systems Journal*, 2023.

[5] M. Alanazi, A. Mahmood, and M. J. M. Chowdhury, "SCADA vulnerabilities and attacks: A review of the state-of-the-art and open issues," *Computers & Security*, vol. 125, p. 103028, 2023.

[6] D. Hasselquist, A. Rawat, and A. Gurtov, "Trends and Detection Avoidance of Internet-Connected Industrial Control Systems," *IEEE Access*, vol. 7, 2019.

[7] M. Cook, A. Marnerides, C. Johnson, and D. Pezaros, "A Survey on Industrial Control System Digital Forensics: Challenges, Advances and Future Directions," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 3, pp. 1705–1747, 2023.

[8] "Threat landscape for industrial automation systems. Statistics for H1 2023," Website. [Online]. Available: https://ics-cert.kaspersky.com/publications/reports

[9] E. Goncharov. (2024) ICS and OT threat predictions for 2024. Head of Kaspersky ICS CERT. [Online]. Available: https://ics-cert.kaspersky.com/publications/reports/2024/01/31/ics-and-ot-threat-predictions-for-2024/

[10] L. Izhikevich, R. Teixeira, and Z. Durumeric, "Predicting IPv4 services across all ports," in *Proceedings of the ACM SIGCOMM 2022 Conference*. New York, NY, USA: Association for Computing Machinery, 2022, p. 503–515.

[11] B. Ren, D. Guo, Y. Yuan, G. Tang, W. Wang, and X. Fu, "Optimal Deployment of SRv6 to Enable Network Interconnection Service," *IEEE/ACM Transactions on Networking*, vol. 30, no. 1, pp. 120–133, 2022.

[12] K. Gaur, A. Kalla, J. Grover, M. Borhani, A. Gurtov, and M. Liyanage, "A Survey of Virtual Private LAN Services (VPLS): Past, Present and Future," *Computer Networks*, vol. 196, p. 108245, 2021.

[13] M. Borhani, I. Avgouleas, M. Liyanage, and A. Gurtov, "KDC Placement Problem in Secure VPLS Networks," *IEEE Transactions on Information Forensics and Security*, vol. 18, pp. 1951–1962, 2023.

[14] M. Bateni, A. Gerber, M. Hajiaghayi, and S. Sen, "Multi-VPN Optimization for Scalable Routing via Relaying," *IEEE/ACM Transactions on Networking*, vol. 18, no. 5, pp. 1544–1556, 2010.

[15] C. Kim, A. Gerber, C. Lund, D. Pei, and S. Sen, "Scalable VPN routing via relaying," in *Proceedings of the 2008 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*. New York, NY, USA: Association for Computing Machinery, 2008, p. 61–72.

[16] T. R. Henderson, S. C. Venema, and D. Mattes, "HIP-based Virtual Private LAN Service (HIPLS)," Internet Engineering Task Force, Internet-Draft draft-henderson-hip-vpls-11, Aug. 2016, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-henderson-hip-vpls/11/

[17] M. Liyanage, J. Okwuibe, M. Ylianttila, and A. Gurtov, "Secure Virtual Private LAN Services: An overview with performance evaluation," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, 2015, pp. 2231–2237.

[18] R. H. Paine, *Beyond HIP: The End to Hacking As We Know It*. BookSurge Publishing, 2009.

[19] "Cisco VPLS Project," Website, 2019. [Online]. Available: https://www.cisco.com/c/en/us/products/ios-nx-os-software/virtual-private-lan-services-vpls

[20] "Vodafone VPN Services," Website, 2025. [Online]. Available: https://www.vodafone.co.uk/business/connectivity/data-connectivity/ethernet-services

[21] Y. Rekhter and K. Kompella, "Virtual Private LAN Service (VPLS) Using BGP for Auto-Discovery and Signaling," Internet Engineering Task Force, Internet-Draft draft-ietf-l2vpn-vpls-bgp-08, 2018, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-l2vpn-vpls-bgp/08/

[22] M. Liyanage, M. Ylianttila, and A. Gurtov, "Fast Transmission Mechanism for Secure VPLS Architectures," in *2017 IEEE International Conference on Computer and Information Technology (CIT)*. IEEE, 2017, pp. 192–196.

[23] M. Liyanage and A. Gurtov, "Securing virtual private LAN service by efficient key management," *Security and Communication Networks*, vol. 7, no. 1, pp. 1–13, 2014.

[24] M. Liyanage, M. Ylianttila, and A. Gurtov, "Software defined VPLS architectures: Opportunities and challenges," in *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, 2017, pp. 1–7.

[25] S. Raghunath, S. Kalyanaraman, and K. Ramakrishnan, "Trade-offs in resource management for virtual private networks," in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, vol. 2, 2005, pp. 1467–1477 vol. 2.

[26] S. Raghunath, K. K. Ramakrishnan, S. Kalyanaraman, and C. Chase, "Measurement Based Characterization and Provisioning of IP VPNs," in *Proceedings of the 4th ACM SIGCOMM Conference on Internet*

Measurement, ser. IMC '04. New York, NY, USA: Association for Computing Machinery, 2004, p. 342–355.
[27] S. L. Hakimi, "Optimum Distribution of Switching Centers in a Communication Network and Some Related Graph Theoretic Problems," *Operations Research*, vol. 13, no. 3, p. 462–475, jun 1965.
[28] O. Kariv and S. L. Hakimi, "An Algorithmic Approach to Network Location Problems. II: The p-Medians," *SIAM Journal on Applied Mathematics*, vol. 37, no. 3, pp. 539–560, 1979.
[29] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)*, first edition ed. W. H. Freeman, 1979. [Online]. Available: http://www.amazon.com/Computers-Intractability-NP-Completeness-Mathematical-Sciences/dp/0716710455
[30] M. L. Fisher, "The Lagrangian Relaxation Method for Solving Integer Programming Problems," *Management Science*, vol. 50, no. 12_supplement, pp. 1861–1871, 2004.
[31] G. Cornuejols, R. Sridharan, and J. Thizy, "A comparison of heuristics and relaxations for the capacitated plant location problem," *European Journal of Operational Research*, vol. 50, no. 3, pp. 280–297, 1991.
[32] L. A. Wolsey, *Integer Programming*, 2nd ed. New York: John Wiley & Sons, 2020.
[33] M. L. Fisher, "The Lagrangian Relaxation Method for Solving Integer Programming Problems," *Management Science*, vol. 50, no. 12_supplement, pp. 1861–1871, 2004.
[34] S. Boyd, L. Xiao, and A. Mutapcic. (2004) Subgradient methods. Lecture notes of EE392o, Stanford University, Autumn Quarter 2004.
[35] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The Internet Topology Zoo," *IEEE Journal on Selected Areas in Communications*, vol. 29, no. 9, pp. 1765–1775, 2011.
[36] S. Orlowski, M. Pióro, A. Tomaszewski, and R. Wessäly, "SNDlib 1.0–Survivable Network Design Library," *Networks*, vol. 55, no. 3, pp. 276–286, 2010. [Online]. Available: http://www3.interscience.wiley.com/journal/122653325/abstract
[37] M. Liyanage and A. Gurtov, "A scalable and secure VPLS architecture for provider provisioned networks," in *2013 IEEE Wireless Communications and Networking Conference (WCNC)*, 2013, pp. 1115–1120.
[38] M. Liyanage, M. Ylianttila, and A. Gurtov, "Improving the tunnel management performance of secure VPLS architectures with SDN," in *2016 13th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2016, pp. 530–536.
[39] K. Mahmood, M. N. Fatima, S. Shamshad, Z. Ghaffar, A. K. Das, and M. J. F. Alenazi, "A Cost-Effective Key Agreement Encryption Protocol for Securing IIoT-Enabled WSN Communication," *IEEE Internet of Things Journal*, vol. 12, no. 5, pp. 5185–5193, 2025.
[40] W. Zhang, H. Zhang, L. Fang, Z. Liu, and C. Ge, "A Secure Revocable Fine-Grained Access Control and Data Sharing Scheme for SCADA in IIoT Systems," *IEEE Internet of Things Journal*, vol. 9, no. 3, pp. 1976–1984, 2022.
[41] A. A. Fröhlich, L. P. Horstmann, and J. L. C. Hoffmann, "A Secure IIoT Gateway Architecture based on Trusted Execution Environments," *Journal of Network and Systems Management*, vol. 31, no. 2, Feb. 2023.
[42] Z. Zhang, W. Huang, Y. Huang, Y. Liao, Z. Zhang, and S. Zhou, "A Domain Isolated Tripartite Authenticated Key Agreement Protocol With Dynamic Revocation and Online Public Identity Updating for IIoT," *IEEE Internet of Things Journal*, vol. 11, no. 9, pp. 15 616–15 632, 2024.
[43] L. Shif, F. Wang, and C.-H. Lung, "Improvement of security and scalability for IoT network using SD-VPN," in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, 2018, pp. 1–5.
[44] A. Atutxa, J. Astorga, M. Barcelo, A. Urbieta, and E. Jacob, "Improving efficiency and security of IIoT communications using in-network validation of server certificate," *Computers in Industry*, vol. 144, p. 103802, 2023.
[45] V. Gugueoth, "LPMLP-Based Framework for Secure IPsec VPN Cloud Gateway with Advanced Network Monitoring and Issue Resolution," in *2023 IEEE 12th International Conference on Cloud Networking (CloudNet)*, 2023, pp. 18–26.
[46] H. Yu, H. Zhang, Z. Yang, and S. Yu, "Edasvic: Enabling Efficient and Dynamic Storage Verification for Clouds of Industrial Internet Platforms," *IEEE Transactions on Information Forensics and Security*, vol. 19, pp. 6896–6909, 2024.
[47] Cisco. (2012) H-VPLS PE Redundancy for QinQ and MPLS Access .
[48] M. Bateni, A. Gerber, M. Hajiaghayi, and S. Sen, "Multi-VPN Optimization for Scalable Routing via Relaying," in *IEEE INFOCOM 2009*, 2009, pp. 2756–2760.
[49] S. Raghunath, K. K. Ramakrishnan, and S. Kalyanaraman, "Measurement-Based Characterization of IP VPNs," *IEEE/ACM Transactions on Networking*, vol. 15, no. 6, pp. 1428–1441, 2007.