PathProb: Probabilistic Inference and Path Scoring for Enhanced and Flexible BGP Route Leak Detection

Yingqian Hao*†, Hui Zou*†, Lu Zhou*†, Yuxuan Chen*† and Yanbiao Li*†✉
*Computer Network Information Center, Chinese Academy of Sciences
†University of Chinese Academy of Sciences
{yqhao, zouhui, chenyuxuan, lybmath}@cnic.cn, zhoulu25@mails.ucas.ac.cn

***Abstract*—The Border Gateway Protocol (BGP) lacks inherent security, leaving the Internet vulnerable to severe threats like route leaks. Existing detection methods suffer from limitations such as rigid binary classification, high false positives, and sparse authoritative AS relationship data. To address these challenges, this paper proposes PathProb—a novel paradigm that flexibly identifies route leaks by calculating topology-aware probability distributions for AS links and computing legitimacy scores for AS paths. Our approach integrates Monte Carlo methods with an Integer Linear Programming formulation of routing policies to derive these solutions efficiently.**

We comprehensively evaluate PathProb using real-world BGP routing traces and route leak incidents. Results show our inference model outperforms state-of-the-art approaches with a high-confidence validation dataset. PathProb detects real-world route leaks with 98.45% recall while simultaneously reducing false positives by $4.29 \sim 20.08$ percentage points over state-of-the-art alternatives. Additionally, PathProb's path legitimacy scoring enables network administrators to dynamically adjust route leak detection thresholds—tailoring security posture to their specific false alarm tolerance and security needs. Finally, PathProb offers seamless compatibility with emerging route leak mitigation mechanisms, such as Autonomous System Provider Authorization (ASPA), enabling flexible integration to enhance leak detection capabilities.

I. Introduction

The Border Gateway Protocol (BGP) is the backbone of the inter-domain routing system, gluing more than $80,000$ Autonomous Systems (ASes) together and underpinning the seamless connectivity we rely on daily [1]. However, BGP has a critical flaw: it lacks built-in security mechanisms to certify and verify routing announcements [2]. This weakness makes it vulnerable to a variety of attacks, including prefix hijacking, path manipulation, and route leaks [3], [4], which can disrupt global Internet security and stability.

✉ Yanbiao Li is the corresponding author.

Among the three types of BGP security incidents, path-manipulation attacks have the lowest incidence and rarely pose actual threats [5]. According to Qrator Labs' BGP incidents report for Q3 2025 [6], $5,937$ unique leakers and $22,967$ hijackers launched route leaks and prefix hijackings during the period, respectively. Although prefix hijackings are the most frequent, standardized defense mechanisms like Resource Public Key Infrastructure (RPKI) [7]-based Route Origin Validation (ROV) [8] and Internet Routing Registry (IRR) [9] have been widely deployed, effectively detecting and mitigating such attacks. Route leaks, by contrast, still lack a widely deployed countermeasure. The only standardized mechanism, Only-To-Customer (OTC) [10], is still in its infancy and unsupported by most routers.[1] In particular, global route leaks (those affecting large numbers of prefixes and ASes, as well as the extent of the anomaly's propagation) occur much more frequently than BGP global hijacks according to Qrator Labs [12], at least since 2021. This means that in the landscape of BGP security defenses, route leaks represent the gap and the most urgent area to address.

A leaked route, though originating from a legitimate AS and traversing an unaltered path, violates routing propagation policies dictated by AS business relationships. The consequences of such violations can be severe, including network outages and economic losses. A vivid event occurred in March 2024: Russian mobile operator MTS (AS8359) leaked tens of thousands of BGP routes learned from the Hong Kong Internet Exchange (AS4635), disrupting global connectivity, especially to Hong Kong, Indonesia, and Australia [13].

Existing route leak detection approaches can be broadly classified into two categories: rule-based white-box approaches and machine learning-driven black-box approaches. The white-box methods [14]–[22] focus on AS links, and use known AS business relationships and the "valley-free rule" [23], a basic principle of how routes should flow, to figure out if a leak is happening. The black-box methods [24]–[30], on the other hand, look at the entire AS path, namely the full sequence of ASes a route passes through. Instead of hard rules, they use

[1]The first commercial implementation—HPE Juniper Networking supports OTC since August 2025 [11].

Network and Distributed System Security (NDSS) Symposium 2026
23-27 February 2026, San Diego, CA, USA
ISBN 979-8-9919276-8-0
https://dx.doi.org/10.14722/ndss.2026.241691
www.ndss-symposium.org

machine learning to learn what normal and abnormal paths look like, then spot leaks based on unusual patterns in those paths. Although black-box techniques can effectively identify complex and novel anomaly patterns, they face significant challenges in practical deployment. First, their decision-making process lacks interpretability—i.e., it is difficult to explain why a specific path is flagged as a leak—and they require substantial volumes of high-quality training data. Furthermore, the complexity of model validation and potential biases in training datasets further hinder their widespread adoption. Consequently, network operators favor white-box methods, which offer high interpretability, transparent decision-making logic, and greater suitability for routine operational scenarios.

However, the efficacy of white-box mechanisms is fundamentally constrained by the confidentiality and sensitivity of AS business relationships [14], [31], [32]—no public dataset exists for these relationships. Existing approaches [14]–[16], [19], [21], [22] therefore rely on inferring deterministic AS relationships from historical BGP routing data, then using these inferred relationships to detect route leaks combined with the valley-free rule. While these inference algorithms achieve reasonable accuracy on best-effort validation sets, they suffer from a high false positive rate in practice, which is primarily due to two main factors: First, AS business relationships are inherently complex, often involving hybrid types or dynamic adjustments driven by routing policies [33]–[35]. Rigid classifications fail to capture this complexity or quantify inference uncertainty, leading to misjudgments. Second, a non-negligible proportion of legitimate paths—stable over time due to commercial interests or policy optimizations—naturally violate the strict valley-free rule [33], [36], [37]. These valid exceptions introduce noise into deterministic inference, further increasing false positives. Despite the existence of ASPA [38]—the only mechanism supporting authoritative declared AS business relationships, built on RPKI and proposed in 2018—it remains unstandardized after 20 iterations and is still in the IETF working group draft stage. Currently, only over 300 ASPA records have been issued [39]. Due to the confidentiality and sensitivity of AS business relationships, authoritative data is expected to remain scarce in the foreseeable future.

To address these challenges, we propose `PathProb`, a "gray-box" framework that detects route leaks by inferring probabilistic AS relationships and computing legitimate scores for AS paths. As a middle ground between white-box and black-box methods, `PathProb` integrates their strengths while avoiding their limitations. On one hand, it retains white-box valley-free validation (ensuring interpretability) but replaces rigid relationship labels with probabilistic distributions to model AS relationship likelihoods. This accommodates the complexity of real-world AS ties (e.g., hybrid, dynamic) and avoids the oversimplification of traditional white-box methods. On the other hand, unlike black-box approaches that binary classify paths as "legitimate" or "leaked," `PathProb` computes mathematical expectations for AS paths based on these probabilistic relationships—reducing "black-or-white" misjudgments by balancing flexibility with transparency in uncertain scenarios. Specifically, `PathProb` assigns a probability distribution to each AS link, then propagates and combines these distributions along the entire AS path to produce a continuous legitimacy score, such as "85% likely to be a leak," instead of a rigid binary label (legitimate or leaked). This continuous output grants `PathProb` significant flexibility: operators can set detection thresholds to align with their network environment and risk tolerance, fine-tuning sensitivity against false-positive rates without re-engineering the system.

Moreover, `PathProb` can seamlessly integrate with ASPA: it adopts authoritative AS relationships declared in ASPA objects where available, and falls back to probabilistic relationships for links without ASPA coverage. In the current data-sparse stage, such probabilistic supplementation immediately boosts leak detection accuracy. This creates a positive feedback loop: improved protection incentivizes more operators to publish ASPA data, gradually enriching the RPKI system with authoritative relationships. As the coverage of authoritative data grows, the increasing authority of the data itself enhances `PathProb`'s gray-box operation—delivering increasingly accurate and robust protection against route leaks.

Overall, our work makes the following key contributions:

- We propose a novel algorithm for inferring probabilistic AS business relationships in Section IV. The algorithm categorizes AS links into two groups based on their topological positions and applies specific inference strategies to each group. To capture the routing path pattern, we develop an Integer Linear Programming (ILP) model, which is the first of its kind in this field. Our algorithm achieves an accuracy of at least 95%.
- We introduce `PathProb` in Section III, a gray-box route leak detection framework that can integrate other AS relationships, such as authoritative ASPA data with inferred probabilistic data. It computes the expected legitimacy score of a route leak, surpassing the limitations of traditional binary classification. In terms of real-world route leak incidents reported by Cloudflare [40], `PathProb` achieves a recall rate of 98.45% by detecting all leaked paths, while maintaining a low false positive rate of 4.17%, which is $4.29 \sim 20.08$ percentage points lower than that of other existing approaches.
- We conduct extensive large-scale simulations to evaluate the performance of `PathProb` when integrated with ASPA across various deployment scenarios in Section V. The results show that even with only 25% deployment rate and no ASPA objects issued, `PathProb` reduces the leakage infection rate (*LIR*) by over 56% (dropping from 15.17% to 6.63%). This demonstrates `PathProb`'s value as a pragmatic solution in the current data-sparse era: it provides immediate defense that effectively bridges the security gap, affording the global routing ecosystem time to incrementally transition toward full authoritative data coverage.

We have open-sourced the `PathProb` implementation to https://doi.org/10.5281/zenodo.17920056 and https://github.com/hyq8868/PathProb.

II. Background and Preliminaries

A. BGP and AS Relationship

The Internet is a large-scale, decentralized network consisting of more than $80,000$ ASes. These ASes interconnect and exchange routing information using BGP, facilitating global end-to-end communication. When an AS exports a route to its neighbors, it appends its own AS number to the `AS_PATH` attribute of the exported route. This process ensures that the `AS_PATH` reflects the sequence of ASes that the route has traversed. In the AS path, each pair of adjacent ASes forms an AS link, and each AS link has a specific type of relationship. The most common types of AS relationships can be categorized as follows:[2]

- **Provider-to-Customer (*p2c*):** AS u provides transit services to AS v, and AS v pays AS u for connectivity.
- **Customer-to-Provider (*c2p*):** The reverse of *p2c*, where AS u is the customer of AS v.
- **Peer-to-Peer (*p2p*):** AS u and AS v exchange traffic freely, usually when they are of comparable size and market position.

The business relationships between ASes define both the economic incentives and the technical policies, which together govern BGP route propagation and traffic exchange.

B. Routing Policy and Valley-free

According to the widely adopted Gao-Rexford model [35], which establishes stability conditions for inter-AS routing based on commercial relationships, each AS follows the export policies listed below:

- An AS exports all routes (including its own, customer-learned, and provider/peer-learned routes) to its customers.
- An AS exports only customer-learned routes to its peers and providers.

If all ASes strictly adhere to the policies defined by the Gao-Rexford model when exporting routes, the resulting AS paths will present a specific topological pattern: AS paths "ascend" through *c2p* links, may optionally include a single horizontal *p2p* link, and then "descend" through *p2c* links. This pattern indicates that AS paths will never transition laterally or back upstream after descending, thereby ensuring that the paths satisfy the valley-free property [23]. The valley-free property imposes specific structural constraints on the sequence of AS links along a path.

- *c2p* links must appear before the *p2c* or *p2p* links.
- *c2p* or *p2p* links must appear before the *p2c* links.
- A path must not contain more than one *p2p* link.

[2]It should be noted that there still exist some special AS relationships, such as sibling ASes that are under shared administrative control and hybrid interconnection models that feature prefix-specific policies or backup links. However, due to their limited prevalence and unique structural characteristics, these special relationships are excluded from the analytical scope of this study.

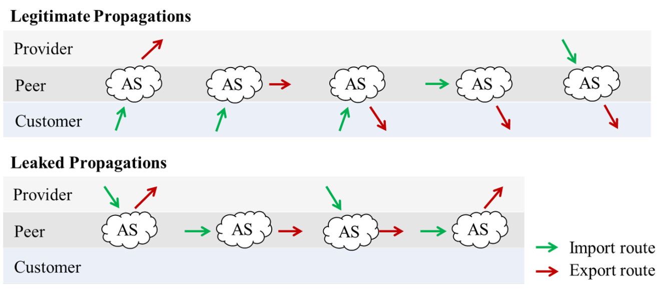


Fig. 1. Legitimate and leaked propagation of routes.

The valley-free property was theoretically established in [23], and further formalized in [41]. Specifically, a path is deemed valley-free if it conforms to one of the following two forms, where $N, M \geq 0$:

- Type-1: N consecutive *c2p* links followed by M consecutive *p2c* links.
- Type-2: N *c2p* links, followed by a single *p2p* link, followed by M *p2c* links.

Fig. 1 depicts five routing patterns that adhere to the above export policies (upper), thereby ensuring that the AS paths exhibit the valley-free property. Conversely, the four patterns (bottom) violate these constraints, constituting route leaks—a phenomenon analyzed in detail in Section II-C.

C. Four Types of Route Leaks

A route leak occurs when an AS propagates routes to unexpected neighbors in violation of established export policies. RFC 7908 [42] categorizes route leaks into six types. Two of these involve tampering with the origin AS and can be prevented by origin validation mechanisms such as RPKI-based ROV [8]. Therefore, our discussion is restricted to the following four types of route leaks.

- Type-1: A route learned from a provider is propagated to another provider.
- Type-2: A route learned from a peer is propagated to another peer.
- Type-3: A route learned from a provider is propagated to a peer.
- Type-4: A route learned from a peer is propagated to a provider.

III. `PathProb`: A Probability-Based Framework for Path Legitimacy Scoring and Route Leak Detection

In this section, we present an overview of `PathProb`, a gray-box route leak detection framework. Assuming that we have obtained the dataset containing the probability distributions of business relationships for each AS link (Section IV), we detail the computation of a legitimacy score for a given AS path—enabling determination of whether a route leak has occurred.

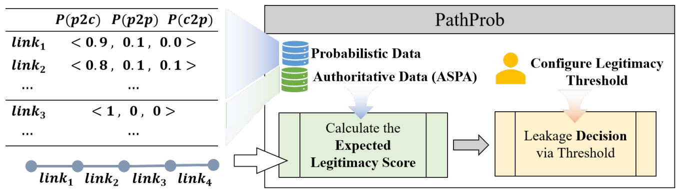


Fig. 2. Overview of `PathProb` architecture.

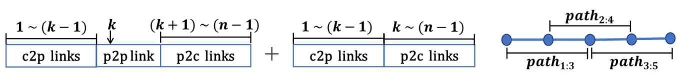


Fig. 3. Calculating the legitimacy score of an AS path.

A. `PathProb` Overview

Fig. 2 illustrates the architecture of `PathProb`, which relies on two sources of AS relationship data: the **authoritative declared** relationship data extracted from ASPA records and the **probabilistic inferred** relationship data derived from BGP routes. Unlike **deterministic inferred** AS relationships, each AS link (u, v) in the probabilistic inferred relationships is represented as a three-dimensional vector, where each dimension corresponds to the probability of one type of relationship, and the sum of the three probabilities equals one. To maintain consistency, each AS link in the authoritative declared relationship data is also converted into a three-dimensional vector, whose entry corresponding to the declared business relationship is set to 1. In contrast, the remaining two entries are set to 0.

Given a route, `PathProb` first extracts `AS_Path` attribute from it and computes the expected legitimacy value of the path based on the aforementioned relationship data. A configurable threshold is then applied: paths whose score is greater than or equal to the threshold are considered **legitimate**, while those with a score below the threshold are classified as **leaked**.

B. Calculating the Legitimacy Score for Each AS Path

Taking the probability distribution of each AS link as input, we propose a method to compute the expected legitimacy of an AS path. In the context of route-leak detection, a path is deemed legitimate if and only if it satisfies the valley-free property. Therefore, the expected legitimacy of an AS path can be computed using the law of total probability by summing the probabilities of all combinations of relationship types that satisfy valley-free constraints.

For a path consisting of n ASes, hence $n - 1$ links, let the relationship of the i-th link be represented by three probabilistic events: $c2p_i$, $p2p_i$, $p2c_i$, then, the expected legitimacy E_{leg} of the path is defined as:

$$\begin{aligned} &E_{\text{leg}}(\text{path}) \\ &= \sum_{k=1}^{n-1} \left(\prod_{i=1}^{k-1} P(c2p_i) \cdot P(p2p_k) \cdot \prod_{i=k+1}^{n-1} P(p2c_i) \right) \\ &+ \sum_{k=1}^{n} \left(\prod_{i=1}^{k-1} P(c2p_i) \cdot \prod_{i=k}^{n-1} P(p2c_i) \right) \end{aligned} \quad (1)$$

As illustrated in Fig. 3, the first term accounts for valley-free paths that contain exactly one *p2p* link. All legitimate (valley-free) cases are enumerated by iterating over every possible position of this *p2p* link within the path. The second term encompasses all valley-free paths that contain no *p2p* links, where legitimate cases are identified by locating the first *p2c* link that initiates the descending segment of the path.

However, the naïve path-level legitimacy expectation compounds the per-link uncertainty, leading to longer paths suffering from systematically lower scores and being easily misclassified as leaks. Intrinsically, a route leak is local: it involves only three consecutive ASes, where the *i*-th AS wrongly exports to the (*i–1*)-th AS a route learned from the (*i+1*)-th AS. This locality implies that the anomaly is independent of the rest of the path. Consequently, we redefine the legitimacy expectation of an AS path as the **minimum** of the expectations computed over all AS triples. This confines the assessment to localized anomalies and neutralizes the distortion introduced by cumulative uncertainty along long paths. After simplification, for a path with *n* ASes, the path-level expectation is given by:

$$\begin{aligned} E_{\text{leg}}(\text{path}) &= \min_{i=1}^{n-2} (E_{\text{leg}}(\text{path}_{i:i+2})) \\ &= \min_{i=1}^{n-2} (P(c2p_i) + P(p2c_{i+1}) - P(c2p_i) \times P(p2c_{i+1})) \end{aligned} \quad (2)$$

In Appendix A, we compare the two scoring schemes with long AS paths, demonstrating that the triple-minimum score is significantly less sensitive to path length than the full-path score.

As shown in Table I, the path is flagged as a route leak because it contains two consecutive *p2p* links with the white-box approach utilizing the `CAIDA` AS relationship data [18], which are deterministically inferred and assign each AS link a single, mutually exclusive relationship type (one-label-per-link). In contrast, our gray-box approach assigns each link a probability distribution over the three relationship types (three-probabilities-per-link) and defines the legitimacy expectation of the path as the minimum expected value among all AS triples, taken only over those triples that satisfy the valley-free constraint. Therefore, even though the relationship type assigned to each AS link under our probabilistic model—taken as the one with the highest probability—matches that of the `CAIDA` dataset, the model yields an expected legitimacy score of 0.697 for the example, allowing this path (observed consistently for six months) to be correctly labeled as legitimate and thereby reducing the false positives produced by the deterministic inferred data. In summary, the probabilistic model abandons the requirement that each link carry a single, fixed relationship; instead, it assigns a probability distribution to every link. Rather than forcing a binary verdict of "legitimate" or "leaked," the model treats legitimacy as a continuous score: a path is accepted as legitimate provided that the **least-legitimate** valley-free triple exceeds the legitimacy threshold,

TABLE I
AN EXAMPLE OF DETERMINING WHETHER A PATH IS LEAKED WITH DETERMINISTIC OR PROBABILISTIC AS RELATIONSHIPS.

Path		202365 - 50673 - 6939 - 199524 - 58212 - 13627					Result
		$link_1$	$link_2$	$link_3$	$link_4$	$link_5$	
White-box		*c2p*	*p2p*	*p2p*	*p2c*	*p2c*	*leaked*
Gray-box	*c2p*	**0.944**	0.451	0.001	0.004	0.0	$E_{\text{leg}} = 0.697$
	p2p	0.056	**0.532**	**0.551**	0.166	0.0	
	p2c	0.0	0.0	0.448	**0.830**	**1.0**	

thereby replacing hard labels with a soft, probability-based criterion and significantly reducing false positives.

IV. PROBABILISTIC AS RELATIONSHIP INFERENCE

In this section, we present our algorithm for inferring probabilistic AS relationships from a set of AS paths described in Section V-A1. We first outline the key challenges, then model the valley-free property as an ILP problem, serving as a tool for inference. Finally, we categorize AS links into two groups: core links and edge links, applying tailored inference strategies to each.

A. Challenges of Inferring Probabilistic AS Relationships

Inferring the probabilities of AS relationships confronts two main challenges.

First, the valley-free principle constitutes the foundational premise for inferring AS relationships. Yet, it presents a twofold dilemma: assigning AS relationships to maximize the number of valley-free paths, namely the Type-of-Relationship (ToR) problem, has been proven NP-complete [17]. Meanwhile, real BGP routes contain many legitimate paths that persistently violate the rule, so minimizing violations alone distorts reality.

To address this, we formulate the valley-free constraints on AS paths (in Section II-B) as an ILP problem. Each AS link's relationship is encoded as a binary integer variable; precedence and cardinality restrictions among *p2c*, *c2p*, and *p2p* links are expressed as linear inequality constraints; and a linear objective function, tailored to the specific inference objective, completes the ILP specification. The general paradigm of an ILP problem is:

$$\text{Maximize/Minimize } \mathbf{c}^T\mathbf{x}$$
$$\text{Subject to: } \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x} \in \mathbb{Z}^n$$

We tailor the ILP formulation to the distinct requirements of each inference stage and solve it with Gurobi [43], a state-of-the-art solver that scales to large problem instances; we also verified that using SCIP [44] yields negligible differences.

Second, because the probabilities of AS relationships are continuous, their precise estimation is inherently more difficult than assigning discrete labels; meanwhile, although ProbLink [15] and TopoScope [16] already employ such probabilities, they use them only as internal cues for assigning one of three fixed types, yielding no interpretable, quantitative values. To overcome this challenge, we propose a hierarchical inference algorithm that leverages the AS-level topological characteristics. Core links—centrally located, densely connected, and relationally complex—are inferred with a sophisticated model. Edge links, numerous and straightforward, are processed with a lightweight method. Crucially, the refined core-link probabilities feed back to sharpen edge-link estimates. While our method is closely related to the BGP routing and AS-level topology, prior studies have demonstrated the long-term stability of these structures [45]–[49].

B. ILP Model for Valley-free Constraint

We propose two ILP models to balance accuracy against feasibility. The *Strict Model* strictly encodes the valley-free property as hard constraints, and the *Loose Model* permits a controlled number of valley-free violations (link skips) to prevent infeasibility, at the cost of a slight loss in strict correctness. These models are integrated into the inference algorithm in sections IV-D and IV-E.

1) ***Strict Model:*** In this model, a pair of binary variables (0 or 1) is used to designate a single relationship type for each AS link; a set of hard constraints then forces the entire path to satisfy the valley-free rule.

Variables. For each AS link i, we introduce a pair of binary variables (x_i, y_i) whose joint values uniquely encode one of the three relationship types. The correspondence between variable combinations and relationship types is predefined as follows:

	$x_i = 0$	$x_i = 1$
$y_i = 0$	*c2p*	–
$y_i = 1$	*p2c*	*p2p*

Constraints. The *Strict Model* imposes the following constraints:

Each AS link is assigned exactly one of the three relationship types, uniquely encoded by one of the three predefined (x_i, y_i) pairs, subject to the following constraints:

$$y_i \geq x_i, \ \forall i = 1, ..., i \ max \tag{3a}$$

For any two links between the same AS pair in opposite directions, namely $link_p = (u, v)$ and $link_q = (v, u)$, their assigned relationship types are (x_p, y_p) and (x_q, y_q), respectively and must be logically consistent (i.e., mutual inverses), enforced by the following constraints:

$$x_p = x_q, \ \forall \text{reversed link } (link_p, \ link_q) \tag{3b}$$
$$y_p + y_q - x_p = 1, \ \forall \text{reversed link } (link_p, \ link_q) \tag{3c}$$

Let $link_{n,m}$ denote the m-th link in the n-th path, and the variable combination for it is $(x_{n,m}, y_{n,m})$. The path is deemed valley-free if it satisfies the conditions defined in Section II-B, which are formalized in the ILP model by the following constraints:

- Any *p2p* or *p2c* link in the path must appear after all *c2p* links.

$$y_{n,m} \le y_{n,m+1}, \quad \forall n = 1,...,n\ max \text{ and } \forall m = 1,...,m\ max \tag{3d}$$

- Any *p2c* link in the path must appear after all *c2p* and *p2p* links.

$$y_{n,m} - x_{n,m} \le y_{n,m+1} - x_{n,m+1}, \quad \forall n = 1,...,n\ max \text{ and } \forall m = 1,...,m\ max \tag{3e}$$

- Each path must contain at most one *p2p* link.

$$\sum_m x_{n,m} \le 1,\ \forall n = 1,...,n\ max \tag{3f}$$

Objective Function. The objective function should be tailored to the inference objective and is specified in Section IV-E for inferring edge links.

2) ***Loose Model:*** In practice, the *Strict Model* often becomes infeasible due to persistent valley-free violations in real-world AS paths. To accommodate this, we extend the model so that selected links in a path may be ignored while valley-free constraints are still enforced on the remaining links.

Variables. For each link i, we augment the binary pair (x_i, y_i) with an additional variable z_i. The variable $z_i = 1$ flags the link as skipped and exempts it from valley-free evaluation, whereas $z_i = 0$ keeps the link in force, with (x_i, y_i) encoding its relationship exactly as prescribed by the *Strict Model*.

	$x_i = 0$	$x_i = 1$
$y_i = 0$	$z_i = 0$, *c2p*	$z_i = 1$, skipped
$y_i = 1$	$z_i = 0$, *p2c*	$z_i = 0$, *p2p*

Constraints. The *Loose Model* imposes the following constraints:

Each AS link is constrained to take exactly one of the above four combinations of (x_i, y_i, z_i), denoting either one of the three relationship types or a skipped link.

$$y_i \ge x_i - z_i,\ \forall i = 1,...,i\ max \tag{4a}$$

$$x_i \ge z_i,\ \forall i = 1,...,i\ max \tag{4b}$$

$$y_i + z_i \le 1,\ \forall i = 1,...,i\ max \tag{4c}$$

Similar to the *Strict Model*, reverse links must have mutually inverse relationship types. Moreover, a pair of reverse links, $link_p = (u, v)$ and $link_q = (v, u)$, must be simultaneously skipped or retained; skipping one without the other is prohibited.

$$x_p = x_q,\ \forall \text{reversed link } (link_p,\ link_q) \tag{4d}$$

$$z_p = z_q,\ \forall \text{reversed link } (link_p,\ link_q) \tag{4e}$$

$$y_p + y_q - x_p + 2z_p = 1,\ \forall \text{reversed link } (link_p,\ link_q) \tag{4f}$$

Let $link_{n,m}$ denote the m-th link in the n-th path and the variable combination for it is $(x_{n,m}, y_{n,m}, z_{n,m})$. After removing all skipped links whose $z_{n,m} = 1$, the remaining links in each path must satisfy the valley-free conditions defined in Section II-B, which are formalized in the ILP model by the following constraints. It should be noted that the *Loose Model*'s allowance for skipped links breaks the global transitivity exploited by the *Strict Model*; valley-free constraints are therefore enforced between every ordered pair of retained links.

- Any *p2p* or *p2c* link in the path must appear after all *c2p* links.

$$y_{n,m_1} \le y_{n,m_2} + z_{n,m_2}, \quad \forall n = 1,...,n\ max \text{ and } \forall m_2 = 2,...,m_2\ max \text{ and } \forall m_1 = 1,...,m_2 - 1 \tag{4g}$$

- Any *p2c* link in the path must appear after all *p2p* or *c2p* links.

$$y_{n,m_1} - x_{n,m_1} \le y_{n,m_2} - x_{n,m_2} + 2z_{n,m_2}, \quad \forall n = 1,...,n\ max \text{ and } \forall m_2 = 2,...,m_2\ max \text{ and } \forall m_1 = 1,...,m_2 - 1 \tag{4h}$$

- Each path may contain at most one *p2p* link.

$$\sum_m (x_{n,m} - z_{n,m}) \le 1,\ \forall n = 1,...,n\ max \tag{4i}$$

Objective Function. The objective of the *Loose Model* is to minimize the number of skipped links, permitting valley-free violations only when unavoidable and thereby preserving maximal fidelity to the original path structure.

$$Minimize \sum_i z_i \tag{4j}$$

C. Identification of Core and Edge Links

To reconcile computational tractability with inference accuracy, we deliberately categorize AS links into edge links and core links. Edge links—numerous yet sparsely observed and hierarchically simple—yield to lightweight methods, whereas the fewer but frequently traversed core links—with their complex, ambiguous relationships—demand sophisticated treatment.

We propose an iterative path-endpoint elimination algorithm and apply it to distinguish between edge and core links. The procedure works as follows:

- A link (X, Y) is classified as an **edge link** if it appears exclusively at the path periphery—either X or Y always at the start (as the first hop) or always at the end (as the last hop)—and never in any interior position. As illustrated in Fig.4, such links occur only in Case-1 or Case-2 paths; no instances of Case 3–6 involve them.
- Upon identifying an edge link, we remove the corresponding endpoint (X or Y) and substitute the original path with its trimmed counterpart.
- The procedure iterates: after each round, new edge links are identified and their outermost hops are removed; then,

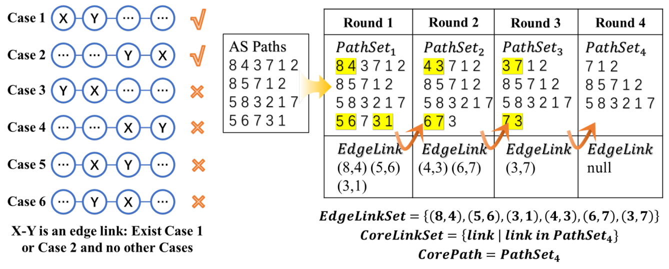


Fig. 4. Identifying core links and edge links.

the corresponding paths are updated with the trimmed ones.

- The iteration terminates when no additional edge links can be identified. At this point, the resulting residual paths, now comprising exclusively core links, are denoted as core paths.

The right side of Fig. 4 provides a running example that walks through the four iterations, showing at each step which edge links are identified. For each iteration, the top row lists the current path set, and the bottom row lists the newly identified edge links. After four rounds, the algorithm terminates, as no more edge links are detected. The complete set of edge links is the union of those identified in each iteration. The remaining links are labeled as core links.

For clarity, we introduce the formal definitions of core links and edge links. Let $P_i = \{p_1, p_2, \ldots, p_m\}$ denote the set of AS paths during the *i*-th iteration, where each path $p_j \in P$ is an ordered tuple $p_j = (a_1, a_2, \ldots, a_k)$, with a_l being the AS at position l. For any path p, its link set is defined as:

$$\text{Link}(p) = \{(a_i, a_{i+1}) \mid i \in \{1, 2, \ldots, k-1\}\}$$

Each link is considered undirected, i.e.,(u, v) and (v, u) represent the same link.

Edge Link Set EL_i. A link (u, v) is an **edge link** if it appears only at the path periphery, with ASu or ASv always at the first or last hop on the path. Let EL_i denote the set of edge links produced upon completion of the i-th iteration, and it can be formalized as:

$$\begin{aligned} EL_i = \big\{ &(u,v) \mid \\ &\big(\forall p \in P_i, (u,v) \in \text{Link}(p) \Rightarrow u \in \{a_1, a_k\}\big) \vee \\ &\big(\forall p \in P_i, (u,v) \in \text{Link}(p) \Rightarrow v \in \{a_1, a_k\}\big)\big\} \end{aligned}$$

Construction Rule from P_i to P_{i+1}. We generate the set P_{i+1} by trimming the peripheral links of each path $p = (a_1, \ldots, a_k) \in P_i$. Specifically, we remove certain links from the start and end of the path. The start offset s and the end offset e are determined by indicator functions $\mathbb{I}(\cdot)$, which check whether the first link (a_1, a_2) and the last link (a_{k-1}, a_k) of path p belong to the edge link set EL_i. If they do, s and e are set to 1; otherwise, they are set to 0. The new path $p_{1+s,k-e}$ is created by removing the first s links and the last e links from the original path p. Applying this trimming process to each path in P_i constructs the new set P_{i+1}.

$$P_{i+1} = \{p_{1+s\,:\,k-e} \mid p = (a_1, \ldots, a_k) \in P_i\}$$

Where the start and end offsets are given by:

$$s = \mathbb{I}((a_1, a_2) \in EL_i) \quad \text{and} \quad e = \mathbb{I}((a_{k-1}, a_k) \in EL_i)$$

Complete Edge Link Set. The process terminates when no further edge links appear, i.e., $EL_n = \emptyset$. The complete edge link set is the union of all EL_i across iterations:

$$\text{EdgeLink} = \bigcup_{i=1}^{\infty} EL_i$$

In practice, the iteration terminates after a finite number of rounds when $EL_i = \phi$, since each iteration strictly shrinks the path set.

Core Links. The remaining links, which are not classified as edge links, are defined as core links:

$$\text{CoreLink} = \bigcup_{p \in P} Link(p) \setminus EdgeLink$$

D. Inferring Core Links

In the previous section, we obtained a set of core links and a set of paths composed exclusively of these core links. To infer the relationship probabilities, it is essential to model the dependency structure among links within paths. Given the complex state space resulting from these dependencies, we adopt a Markov Random Field (MRF) framework, which allows us to model the interactions between links in a probabilistic manner. To efficiently perform inference, we employ Gibbs sampling for approximate inference. This approach enables us to iteratively update the relationship of each link based on the current state of its neighbors, thereby capturing the dependencies while maintaining computational feasibility.

Each core link is modeled as a random variable x_i, which can take one of three values: *p2p*, *p2c*, or *c2p*. Let $X = (x_1, x_2, \ldots, x_n)$ be the vector of all link variables, and let $P(X)$ denote the joint probability distribution over their assignments. Our goal is to estimate the marginal distribution $P(x_i)$ for each link.

The valley-free property implies that the relationship of each link depends only on its neighbors within a path and is conditionally independent of other links. This local dependency structure is well-suited to the definition of an MRF, where each variable x_i is conditionally independent of all others given the states of its neighboring set $N(i)$. In an MRF, the joint probability distribution over all variables can be factorized into a product of potential functions defined over local cliques:

$$P(X) = \frac{1}{Z} \prod_i \phi_i(x_i, x_{N(i)}) \tag{5}$$

$$Z = \sum_X \prod_i \phi_i(x_i, x_{N(i)}) \tag{6}$$

where Z is the partition function (normalization constant) ensuring the distribution $(P(X))$ sums to 1.

Each potential function ϕ_i models the compatibility between the state of x_i and its neighbors. We define ϕ_i based on valley-free routing constraints, which reflect AS export policies.

$$\phi_i(x_i, x_{N(i)}) = \sum_k \alpha_{ik}(x_i, x_i^{\text{prev}(k)}, x_i^{\text{next}(k)}) \tag{7}$$

TABLE II
THE MAPPINGS BETWEEN THE TRIPLE COMPRISING A LINK WITH ITS NEIGHBOR PAIR, AND ITS α VALUE.

x^{prev}	x^{next}	x	α	x^{prev}	x^{next}	x	α
c2p	*c2p*	*c2p*	1	*p2p*	*c2p*	*c2p*	0
c2p	*c2p*	*p2p*	0	*p2p*	*c2p*	*p2p*	1
c2p	*c2p*	*p2c*	0	*p2p*	*c2p*	*p2c*	0
c2p	*p2p*	*c2p*	1	*p2p*	*p2p*	*c2p*	0
c2p	*p2p*	*p2p*	0	*p2p*	*p2p*	*p2p*	1
c2p	*p2p*	*p2c*	0	*p2p*	*p2p*	*p2c*	0
c2p	*p2c*	*c2p*	0	*p2p*	*p2c*	*p2c*	1
c2p	*p2c*	*p2p*	1	*p2p*	*p2c*	*p2p*	0
c2p	*p2c*	*p2c*	0	*p2p*	*p2c*	*p2c*	0
p2c	*c2p*	*c2p*	0	*p2c*	*p2p*	*p2c*	0
p2c	*c2p*	*p2p*	1	*p2c*	*p2c*	*c2p*	0
p2c	*c2p*	*p2c*	0	*p2c*	*p2c*	*p2p*	0
p2c	*p2p*	*c2p*	0	*p2c*	*p2c*	*p2c*	1
p2c	*p2p*	*p2p*	1				

For each link x_i, we examine its k-th neighbor pair $(x_i^{\text{prev}(k)}, x_i^{\text{next}(k)})$, where $x_i^{\text{prev}(k)}$ and $x_i^{\text{next}(k)}$ denote the predecessor and successor links of x_i in its k-th path. If x_i is the first link in a path, we set its predecessor $x_i^{\text{prev}(k)}$ to a default value of *c2p* to avoid restricting x_i's possible values. Similarly, if x_i is the last link, its successor $x_i^{\text{next}(k)}$ is defaulted to *p2c*, again to avoid imposing constraints on x_i due to incomplete context. In addition, α_{ik} defines a compatibility mapping based on the valley-free property. The mapping rules from the triplet $(x^{\text{prev}}, x, x^{\text{next}})$ to α are detailed in Table II. Each element in this tuple can take one of three relationships, resulting in 27 combinations. For a given pair $(x^{\text{prev}}, x^{\text{next}})$, only one of the three tuples has its α set to 1, and the other two tuples have their α set to 0. Tuples with α equal to 1 either satisfy the valley-free principle or x is a *p2p* relationship, minimizing the injection of uncertain or conflicting information into the inference model.

To address the computational intractability arising from the exponential growth of the state space ($O(nk{\cdot}3^n)$) with increasing variables, we employ Gibbs sampling, a Markov Chain Monte Carlo (MCMC) method that efficiently approximates the target distribution. This approach is particularly well-suited for our high-dimensional model, where direct computation of the joint distribution is infeasible, but the local conditional distributions remain computationally tractable.

The Gibbs sampling algorithm generates a Markov chain of samples by iteratively resampling each variable x_i from its conditional distribution $P(x_i \mid X \setminus \{x_i\})$, where $X \setminus \{x_i\}$ denotes the set of all variables except x_i. The conditional probability for each variable x_i is computed as:

$$P(x_i \mid X \setminus \{x_i\}) = P(x_i \mid x_{\mathcal{N}(i)}) = \frac{\phi_i(x_i, x_{\mathcal{N}(i)})}{\sum_{x_i'} \phi_i(x_i', x_{\mathcal{N}(i)})} \quad (8)$$

Here, $\mathcal{N}(i)$ denotes the neighboring variables of x_i, and ϕ_i is the potential function defined earlier.

More specifically, to generate K samples, denoted as $X^{(i)} = (x_1^{(i)}, \ldots, x_n^{(i)})$ for $i = 1, \ldots, K$, the Gibbs sampling procedure proceeds iteratively as follows:

1) **Initialization**: Assign initial values to all variables $X^{(1)} = (x_1^{(1)}, \ldots, x_n^{(1)})$, either randomly or via heuristic methods.
2) **Conditional Sampling**: For each sample $X^{(i)} = (x_1^{(i)}, \ldots, x_n^{(i)})$ (where $i = 1$ to K), compute each $x_j^{(i)}$ (where $j = 1$ to n) in sequence using its conditional probability:

$$P(x_j^{(i)} \mid x_1^{(i)}, \ldots, x_{j-1}^{(i)}, x_{j+1}^{(i-1)}, \ldots, x_n^{(i-1)})$$

   Once all variables x_j have been updated, they collectively form the i-th sample.
3) **Iteration**: The Gibbs sampling procedure iteratively repeats this process until K samples have been generated.

The convergence speed of Gibbs sampling is highly sensitive to the quality of the initial sample. When the initial state lies in low-probability regions of the target distribution, the Markov chain may require extensive iterations to achieve stationarity, increasing computational costs and delaying inference. To mitigate this, we employ the output of our *Loose Model* (Section IV-B2) as a warm start for optimization. By relaxing the valley-free constraints, the model generates a starting point within the high-density regions of the target distribution, thereby accelerating convergence and enhancing inference quality. In our implementation, we sample $K = 1{,}000$ instances for marginal estimation. This choice is guided by a sensitivity analysis (Appendix B-A), which confirms that $K = 1{,}000$ strikes an optimal balance between stability and computational efficiency.

E. Inferring Edge Links

In Internet topology, core links dominate connectivity patterns, while edge links are observed at path boundaries. Due to their infrequent occurrence, edge link relationships can typically be inferred deterministically through propagation from core links. Therefore, we propose a deterministic propagation mechanism based on core links for inferring the relationship of edge links, which mainly consists of two steps: core-to-edge propagation and context-aware resolution.

More specifically, if the last core link in a path has a high probability (e.g., over 80% in our setting) of being either *p2c* or *p2p*, then the subsequent edge link is inferred as *p2c*. Conversely, if the first core link is highly likely to be *c2p* or *p2p*, the preceding edge link is inferred as *c2p*. We conduct a sensitivity analysis of the core-to-edge threshold in Appendix B-B, demonstrating that our results remain robust under this reasonable threshold selection. This inference process can be recursively applied: once an edge link is labeled, its relationship propagates to neighboring links in other paths, leveraging path overlap to resolve ambiguities. As illustrated in Fig. 5, in $path\,1$, link (C, A) propagates its relationship *p2c* to its adjacent edge link (A, B), and the relationship of which can be inferred as *p2c*. This inferred relationship information further propagates to adjacent links in $path\,2$, and the relationship of its neighbor edge link can also be inferred accordingly as *p2c*.

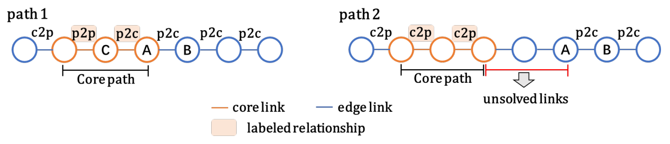


Fig. 5. Edge links relationship inference via recursive contextual propagation.

For the remaining edge links with undetermined relationships, they can be categorized into two types: isolated links, which have no neighbor links, and contextual links, which have at least one neighbor link, each requiring a distinct processing strategy. For isolated edge links, uniform prior probabilities $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ are assigned to *p2p*, *p2c* and *c2p* for an isolated edge link to ensure unbiased inference.

For contextual edge links, we employ the *Strict Model* (see Section IV-B1). Since edge links are constrained to path boundaries, the resulting ILP problem is always solvable. A straightforward, feasible solution is to assign all edge links as *p2c*, treating the outermost ASes as customers. Our objective function is designed to prioritize solutions that maximize the number of paths conforming to the structure $p2p + n \cdot p2c$, which aligns with expected routing patterns in hierarchical networks.

V. Evaluation

In this section, we conduct extensive experiments to evaluate the performance of the `PathProb` framework and verify its superiority in comparison with `AS-Rank`, `ProbLink`, and `TopoScope`.

A. Methodology

1) Data sets: We use the preprocessed AS path dataset to infer AS relationships, and evaluate `PathProb` using the validation datasets.

AS Paths. We collect AS paths from BGP routing data archived by RouteViews [50] and RIPE RIS [51]. The two projects operate multiple collectors that export snapshots of BGP routing tables (*RIBs*) every 2 hours (RouteViews) and 8 hours (RIPE RIS), and BGP *updates* every 15 and 5 minutes, respectively. We infer AS relationships using AS paths extracted from the RIBs and evaluate the performance of four schemes in detecting route leaks using AS paths extracted from updates. To ensure comparability of evaluated schemes, we select the same 33 collectors as [18], listed in Table VIII. For BGP RIB datasets, we use both long-term and short-term RIBs. The dataset `RIB_Year` contains 12 RIB sets, each corresponding to a month from August 2024 to July 2025. We collect and merge the $1,170$ RIB subsets archived by the 33 collectors over the first five days of each month. The dataset `RIB_Day` contains 30 RIB sets from July 1st to 10th, 2025, with each set representing the union of BGP RIBs from the 33 collectors over an 8-hour interval. The details about two AS path datasets can be found in Table VII. BGP update datasets, we focus on June 2025 and use the route leak events reported by Cloudflare Radar [40] together with BGP updates from the same 33 collectors to construct two labeled AS-path datasets: `Update_Event_Tune`, used for threshold selection, and `Update_Event_VD`, used for route leak detection performance evaluation. Details of the `Update_Event_VD` are summarized in Table VIII.

Preprocessing. We preprocess the collected AS paths to filter mistakes and noises that may impact the relationship inference. First, we obtain a list of Internet eXchange Points (IXPs) from PeeringDB and remove their ASNs from the AS paths because they do not participate in the routing. Second, we remove duplicate ASes caused by AS path prepending. Third, we discard paths containing AS loops, which indicate abnormal routing behaviors or unassigned ASNs that should not appear on the global Internet.

Validation Datasets. We employ two types of ground-truth datasets: the *AS relationship* validation dataset and the *route leak* benchmark dataset, to evaluate two aspects of the four schemes, namely the AS relationship inference and route leak detection.

- **AS Relationships.** Due to the privacy and sensitivity of AS relationships, a definitive ground truth is unavailable. To evaluate the performance of the four schemes in AS relationship inference, we utilize two complementary datasets, which together form the dataset `AS_Rel_VD`. The first is the `CAIDA Serial-2` AS relationship dataset [18], the most authoritative and comprehensive publicly available dataset. It integrates various methods to infer AS relationships, including heuristic approaches, traceroute-based inferences (Ark), inferences from BGP communities, and AS relationships obtained from multi-lateral peering repositories, offering extensive global coverage with monthly updates. The second dataset consists of ASPA objects collected by the RPKI relying-party software rpki-client [39]. These objects, voluntarily issued by ASes, ensure high trustworthiness but cover only a few hundred *p2c* relationships. These two validation datasets complement each other in terms of data coverage and precision. The `CAIDA` dataset provides broad coverage, albeit with best-effort precision, while the `ASPA` dataset offers high precision but with a limited scope.
- **Route Leaks.** We evaluate the performance of the four schemes in route leak detection by evaluating their ability to distinguish between legitimate and leaked paths. We use the route leak events reported by `Cloudflare Radar` [40] as the ground truth. Each event includes a timestamp and a leak segment, which is an AS triplet $[ASa, ASb, ASc]$, indicating that ASb incorrectly exported the route from ASc to ASa. First, we determine the time window for each route leak event (5 / 15 minutes before and after the event for RIPE RIS and RouteViews, respectively). Second, we collect BGP updates over a specified time window, extract AS paths from these updates, and label each path as "leaked" or "legitimate" by checking if it contains the leaked segment. Using this method, we generated a labeled dataset of AS paths span-

TABLE III
DEFINITIONS OF AS ROLES AND LINK TYPES USED IN EVALUATION.

Type	Role Definitions & Selection Criteria
AS Roles	**Tier-1 (T1.)**: An AS classified as a "Tier 1 network" per Wikipedia [52]. **Stub (Stub)**: An AS with its degree $d = 1$ in [18]. **Transit (Trans.)**: An AS that acts as a provider to at least one AS and has a degree $d > 1$ in [18]. **Content (Cont.)**: An AS classified as the "Content" type in the PeeringDB [53]. **High Impact (HI.)**: ASes of critical importance in network topology and traffic transmission, typically characterized by large-scale, concentrated traffic and key influence on global/regional routing stability and performance. The AS list was independently generated by our team through online research, and is now publicly hosted in our open-source repository.
Link Types	**Sibling (Sib.)**: The two ASes in the AS link belong to the same organization in [54]. **Partial Transit (PT.)**: The two ASes in the AS link have a "partial-transit" relationship as per [33].[3] **Hybrid (Hyb.)**: The two ASes in the AS link have a "hybrid" relationship as per [33].[3]

ning 30 days in June 2025. The first 15 days constitute the dataset `Update_Event_Tune`, reserved exclusively for threshold selection, while the remaining 15 days form the dataset `Update_Event_VD`, designed as the evaluation dataset for route leak detection. To evaluate `PathProb`'s robustness in a fine-grained, multi-dimensional manner, we partition the dataset `Update_Event_VD` along three axes: (i) **temporal partitioning** with each day as an independent test set to gauge performance consistency over time; (ii) **collector-based grouping**, namely grouping paths by the 33 collectors to quantify potential vantage point bias; and (iii) **network role filtering**, which filters paths traversing specific ASes or links defined in Table III to analyze how diverse AS types and link characteristics affect detection accuracy. For more details, please refer to Appendix C.

2) Evaluated schemes: We evaluate our proposed scheme `PathProb`, along with three other schemes: `AS-Rank`, `ProbLink`, and `TopoScope`. The implementation of `AS-Rank`[4] is obtained from `CAIDA`, while the implementations of `ProbLink`[5] and `TopoScope`[6] are provided by their respective authors.[7]

3) Performance metrics: For the AS relationship reference, we introduce the metric **accuracy** to compare four schemes, which is measured as the ratio of the number of AS links correctly inferred in the AS relationship validation dataset to the total number of AS links that can be inferred in the dataset. For route leak detection, we introduce four concepts that are used to define the three key metrics for evaluating the performance of four schemes. A path is counted as a false positive (#fp) if it is identified as a leaked path but is actually a legitimate path. Similarly, a path is counted as a false negative (#fn) if it is identified as a legitimate path but is actually a leaked path. Conversely, correctly identified leaked paths are classified as true positives (#tp). Correctly identified legitimate paths are classified as true negatives (#tn). Then, the metric **precise** is measured as $\frac{\#tp}{\#tp+\#fp}$, the metric **recall** is given by $\frac{\#tp}{\#tp+\#fn}$, and the metric false positive rate (**FPR**) is defined by $\frac{\#fp}{\#fp+\#tn}$. These three metrics enable a comprehensive evaluation of route leak detection performance. Recall quantifies detection coverage of route leaks, while precision measures the reliability of alerts. The FPR is the most critical metric. Excessively high FPRs have been a major barrier to the adoption of previous solutions. Therefore, the core challenge of our work is how to minimize FPR while maintaining high recall. However, the three metrics are strongly correlated with the distribution of positive and negative samples. As shown in Table VIII, there is a significant imbalance between leaked paths and legitimate paths (positive and negative samples) in the validation dataset. To mitigate the impact of this imbalance on the evaluation results, we adopt a sample weighting method, assigning different weights to positive and negative samples. Specifically, the weight of each class is set to be inversely proportional to the number of samples it contains. This ensures that the fewer positive samples (leaked paths) receive higher weights. In contrast, the more negative samples (legitimate paths) are assigned lower weights, thereby balancing their overall contribution to the final performance.

4) Simulation environment: To evaluate the impact of `PathProb` on mitigating the effects of route leaks, we ran simulations using a framework adopted by [55], [56]. We constructed an AS-level global Internet topology using the `CAIDA Serial-1` AS relationship dataset, where all ASes adhere to the standard Gao-Rexford model for routing selection and export. In each simulation run, we set up different deployment scenarios and randomly selected 1,000 AS pairs as leakers and victims. Specifically, the leaker ASes were chosen at random from non-stub ASes, namely those with more than one neighbor, to ensure they have the capacity to propagate leaked routes. The victim ASes were selected randomly, excluding any direct or indirect customer ASes of the corresponding leakers, because an AS can export routes it receives from any type of neighbor to its customers, which would not lead to route leaks. The leaker ASes announced routes originated by the victims to their neighbors. In each route leak event, each AS in the global network topology will fall into one of the following three categories:

- **Immune**: ASes that reach the victim via legitimate paths.
- **Infected**: ASes that reach the victim via leaked paths.

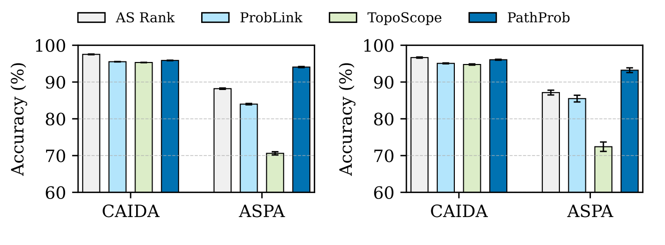
[3]Due to the absence of up-to-date alternatives, we draw partial-transit and complex relationships from the 2014 `CAIDA` supplement dataset, treating them as approximate indicators of link types.

[4]https://publicdata.caida.org/datasets/as-relationships/2013-asrank-data-extra/

[5]https://github.com/YuchenJin/ProbLink

[6]https://github.com/Zitong-Jin/TopoScope

[7]LeMon [22] is open-source but relies on offline analysis of globally aggregated data, differing from our approach and thus not directly comparable, and other white-box approaches mentioned in Section VI are not open-sourced.



(a) `RIB_Day` path dataset (b) `RIB_Year` path dataset

Fig. 6. AS relationship reference accuracy with `RIB_Day` and `RIB_Year`.

- **Disconnected**: ASes that lost all paths to the victim, and cannot reach the victim at all.

Obviously, the more paths identified and filtered as route leaks, the fewer infected ASes but more disconnected ones; fewer filtered paths mean more infected ASes and fewer disconnections. To this end, we define two metrics to evaluate the security and reachability of the four schemes in detecting route leaks. Let N_{inf}, N_{imm}, and N_{disc} denote the numbers of infected, immune, and disconnected ASes, respectively. The one metric *leakage infection rate (LIR)* refers to the proportion of infected ASes among ASes that have paths to the victim and is calculated as $\frac{N_{\text{inf}}}{N_{\text{inf}}+N_{\text{imm}}}$. The other metric *legitimate connection rate (LCR)* refers to the proportion of immune ASes among all ASes and is calculated as $\frac{N_{\text{imm}}}{N_{\text{inf}}+N_{\text{imm}}+N_{\text{disc}}}$.

B. AS Relationship Inference

Figs. 6(a) and 6(b) show the accuracy of four evaluated schemes in inferring AS relationships using path datasets `RIB_Day` and `RIB_Year`, along with the corresponding validation dataset `AS_Rel_VD`. Each bar shows the average accuracy over all snapshots, and the error bars denote the 95% confidence intervals. To facilitate comparison with other deterministic inference schemes, we map the probabilistic distribution of an AS link to the relationship type with the highest probability. Overall, when evaluated on the `CAIDA` validation dataset, `PathProb` slightly underperforms `AS Rank` but outperforms `ProbLink` and `TopoScope` in average accuracy for AS relationship inference. On the `RIB_Day` and `RIB_Year` path datasets, `PathProb` achieves average accuracies of 95.87% and 96.06%, respectively, trailing `AS Rank` by merely 1.65 and 0.59 percentage points. This narrow gap is expected, since the `CAIDA` validation dataset itself is inferred from an optimized solution of `AS Rank`, it inherently favors `AS Rank`-style heuristics, creating a natural baseline advantage for that method. However, when employing the `ASPA` validation dataset, which has the highest credibility in AS relationships, `PathProb` achieves the best performance in AS relationship inference average accuracy, outperforming other schemes by 5.87 $\sim$ 23.46 and 6.08 $\sim$ 20.81 percentage points using `RIB_Day` and `RIB_Year`, respectively. The 95% confidence intervals in Fig. 6 are non-overlapping, indicating statistical significance of these performance gains. We hypothesize that other schemes exhibit suboptimal performance on the *p2c*-rich `ASPA` data, likely due to a tendency to infer neutral *p2p* relationships. By contrast, `PathProb` more effectively captures these dynamics through its probabilistic modeling framework.

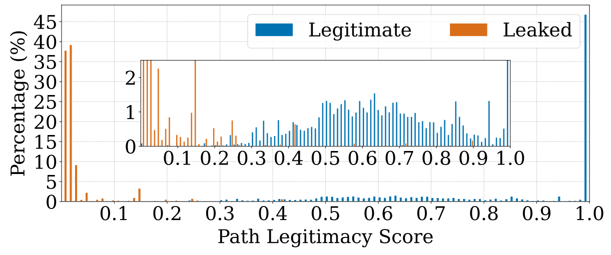


Fig. 7. The legitimacy score distribution of legitimate and leaked paths.

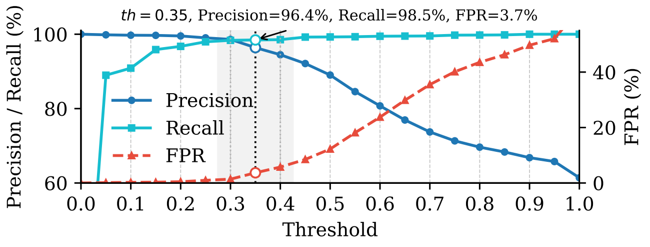


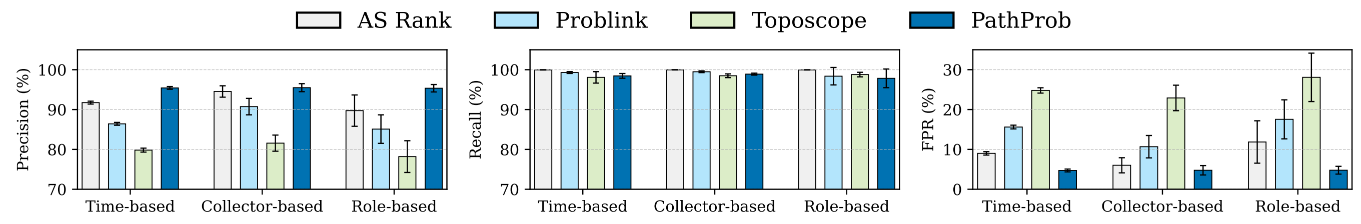
Fig. 8. The impact of threshold on *precision*, *recall*, and *FPR*.

C. Route Leak Detection

1) Path score distributions with `PathProb`: For each link, we calculate a legitimate score based on the inferred relationship probabilities with `PathProb` following the method outlined in Section III-B. If the score is below the threshold th, the path is flagged as a potential leak; otherwise, it is deemed legitimate. Fig. 7 shows the score distributions of legitimate and leaked paths in the dataset `Update_Event_Tune`.[8] There is a significant difference in the scores of the two types of paths. The vast majority of leaked paths have scores concentrated between 0 and 0.1, close to zero, indicating that `PathProb` identifies them as highly suspicious leaked paths. Conversely, legitimate paths exhibit significantly higher scores: 46.83% of them have scores exceeding 0.99, and another 37.45% have scores ranging from 0.5 to 0.9.

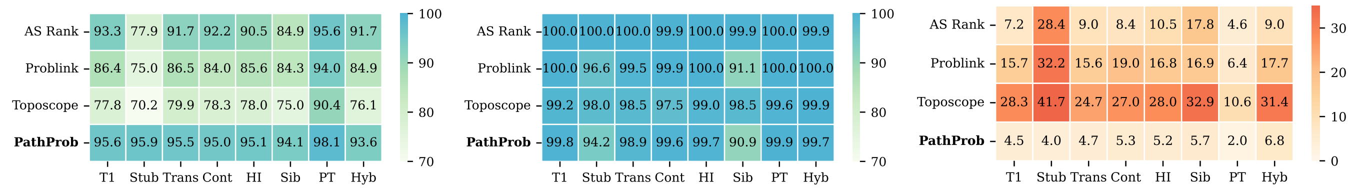
2) Configuration of `PathProb`: However, whether a suspicious route leak is determined to be a route leak depends on whether its score is below the threshold. Therefore, selecting a threshold requires balancing *precision*, *recall*, and *FPR*. As shown in Fig. 8, we evaluate the three metrics across a range of threshold values. Ultimately, we set the threshold at 0.35. At this threshold, `PathProb` achieves a high *recall* (96.4%), ensuring that nearly all leaked routes are detected. Meanwhile, this threshold maintains a high *precision* (98.5%) and keeps the *FPR* relatively low (3.7%), providing a trade-off between detection coverage and false alarm rate.

[8]Each bar represents a bin of width 0.01, with x-axis labels displayed at intervals of 0.1 units.



(a) Precision (b) Recall (c) FPR

Fig. 9. The *precision*, *recall*, and *FPR* of four schemes in detecting route leaks using validation dataset `Update_Event_VD`.



(a) Precision (b) Recall (c) FPR

Fig. 10. The *precision*, *recall*, and *FPR* of four schemes in detecting route leaks

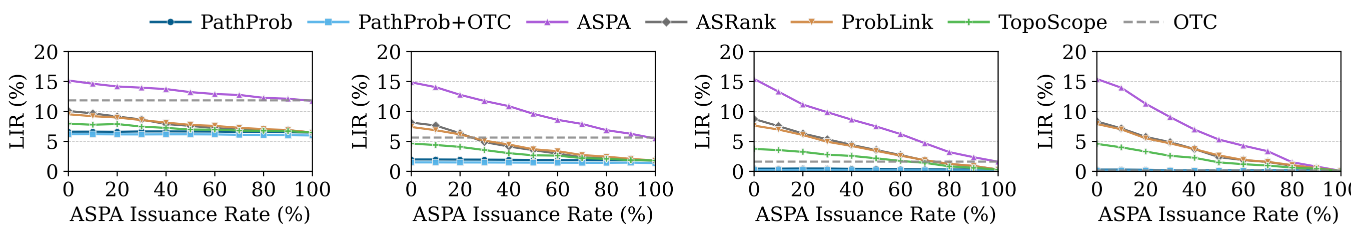
3) Comparison with other schemes: We evaluate four schemes using the validation dataset `Update_Event_VD` across the three dimensions defined earlier—time, collector, and network role—with error bars indicating 95% confidence intervals. Across all three dimensions, `PathProb` demonstrates significant superiority, consistently achieving the highest average *precision* and the lowest *FPR* in Fig. 9. Specifically, in the time dimension, `PathProb` achieves an average *precision* of 95.43%, exceeding the other three schemes by $3.69 \sim 15.61$ percentage points. Crucially, its average *FPR* is merely 4.71%, which is $4.29 \sim 20.08$ percentage points lower than the competing schemes. In the collector dimension, `PathProb` maintains an average *precision* of 95.50%, improving by $0.96 \sim 13.92$ percentage points, while suppressing the average *FPR* to 4.76%—a reduction of $1.25 \sim 18.15$ percentage points. Regarding the network role dimension, `PathProb` outperforms `AS Rank`, `Problink`, and `Toposcope` with an average *precision* of 95.35% (leading by $5.62 \sim 17.15$ percentage points) and a reduced average *FPR* of 4.78% ($7.08 \sim 23.30$ percentage points lower), underscoring its robust performance across diverse evaluation scenarios. Fig. 10 provides a granular breakdown of results for role-specific AS subsets and type-specific link subsets `PathProb` consistently achieves best *precision* and *FPR*, outperforming alternatives in every specialized scenario. These findings confirm that our proposed method exhibits strong robustness to temporal variations, vantage point biases, and heterogeneous network characteristics. In terms of *recall*, the four schemes demonstrate comparable effectiveness across all dimensions. All recall values range from 97.84% to 99.97%, with a maximum divergence of merely 2.13 percentage points. Although `PathProb` is not always the absolute leader in this metric, it sustains a recall rate approaching 100%—aligning with the top-performing baselines and reflecting its reliable detection coverage.

The main reason is that `PathProb` does not simply label a path as leaked or legitimate, but calculates a legitimacy score for each path. `PathProb` not only considers the path as a whole but also provides operators with sufficient flexibility, allowing them to adjust the threshold according to their security needs.

D. Simulation Experiment

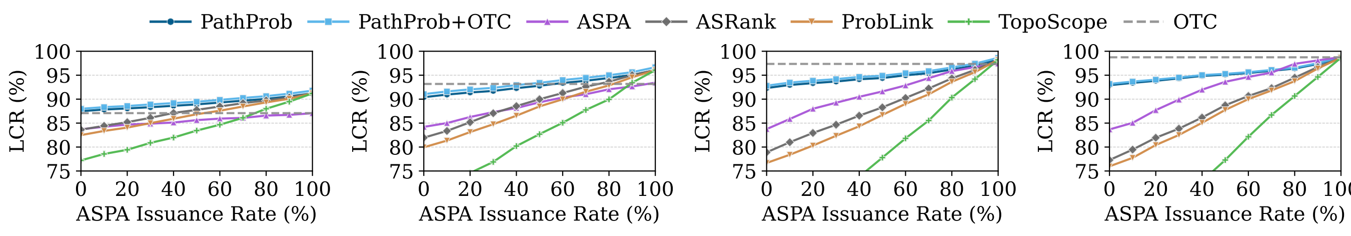
In this section, we evaluate the impact of integrating each of the four schemes with `ASPA` on Internet security and reachability. By configuring various deployment scenarios, we simulate the phased deployment process of `ASPA`. First, we adopt the `ASPA` scheme as the baseline, and then integrate the authoritative declared AS relationships in `ASPA` with those inferred by the other four schemes. Additionally, we incorporate `OTC` [10] as a standalone benchmark and evaluate a hybrid `PathProb+OTC` scheme. In this hybrid approach, a path is identified as a leak if classified as such by either `PathProb` or `OTC`. Finally, by setting up different deployment scenarios, we measure the *LIR* and *LCR* of the five schemes to evaluate their impact on Internet security and reachability once truly deployed. A deployment scenario is defined by both the issuance rate and the deployment rate. The issuance rate refers to the proportion of known AS relationships (including both inferred and declared) out of all AS relationships in the network, with intervals of 10%. In contrast, the deployment rate refers to the proportion of ASes using AS relationships for route leak detection out of all ASes in the network, with intervals of 25%.

As can be seen from Fig. 11 and Fig. 12, `PathProb` and `PathProb+OTC` consistently achieve the lowest *LIR*



(a) 25% deployment (b) 50% deployment (c) 75% deployment (d) 100% deployment

Fig. 11. The *LIR* of four evaluated schemes under different deployment rates (25%, 50%, 75%, and 100%).



(a) 25% deployment (b) 50% deployment (c) 75% deployment (d) 100% deployment

Fig. 12. The *LCR* of four evaluated schemes under different deployment rates (25%, 50%, 75%, and 100%).

and the highest *LCR* across the vast majority of deployment scenarios, demonstrating the robustness of our probabilistic mechanism. Leveraging `OTC`'s complementarity strengths, the hybrid `PathProb+OTC` yields marginally enhanced performance. Crucially, our approach offers significant immediate benefits during early deployment phases. With only 25% deployment and no ASPA issuance (0%), `PathProb` reduces the *LIR* from 15.17% to 6.63% (a 56% decrease) and concurrently improves the *LCR* from 83.70% to 87.49%. The `PathProb+OTC` further optimizes these metrics, reducing *LIR* to 6.19% and boosting *LCR* to 87.99%. As deployment expands, the improvements become increasingly evident—for example, in a scenario with 70% ASPA data issuance and 50% deployment rate, `PathProb` reduces *LIR* down to as low as 1.87% while maintaining a high *LCR* of 93.84%.

Comparatively, other schemes face challenges in balancing security and reachability. `OTC`, rooted in strict RFC-defined rules, introduces zero false positives. While this conservative design maintains the *LCR*, it constrains its leak suppression capability, leading to only marginal *LIR* reductions—particularly in early-stage deployment scenarios. Conversely, deterministic inference schemes (`AS Rank`, `ProbLink`, and `TopoScope`) exhibit aggressive leak reduction capabilities but incur high false-positive rates. Under last-stage deployment scenarios, their *LCR* declines significantly below the `ASPA` baseline, severely compromising network connectivity. By comparison, `PathProb` achieves a superior trade-off between security and reachability. `PathProb`'s competitive results can be attributed to its probabilistic evaluation mechanism. This mechanism assesses the legitimacy score of each path and filters out only those with low confidence. By collectively evaluating all links in a path to determine the likelihood of a route leak, `PathProb` maintains higher overall connectivity. Additionally, `PathProb`'s threshold tuning capability offers flexibility and performance potential. It allows network operators to customize the threshold based on their own network knowledge and operational requirements, enabling better adaptation and improved effectiveness.

E. Deployment model and overhead

In this section, we present a practical deployment model that aligns with the existing ASPA architecture. In this model, the inference of probabilistic AS relationships is conducted out-of-band. Evaluation shows that processing a global BGP snapshot requires less than 2 hours and peaks at 20GB of memory usage, demonstrating the feasibility of regular updates (e.g., monthly intervals). These inferred probabilities are then encapsulated into RPKI objects and distributed via the standard RPKI publication, validation, and distribution protocols. Consequently, existing RPKI software and routers need to be extended to process `PathProb` objects. Routers perform lightweight validation for each incoming BGP update, introducing modest implementation complexity compared to `ASPA`. All operations are confined to the control plane, imposing no additional latency on data-plane traffic.

VI. Related Work

Several studies [24]–[30] have explored machine learning models for route leak detection. These "black-box" approaches detect route leaks directly from routing data without requiring explicit AS relationships. For instance, BEAM [24] introduces a network representation learning model to detect anomalies by capturing AS role changes. RoLL [25] and RoLL+ [26] use AS triplet features for real-time leak detection, achieving high accuracy with low latency. FL-RLD [27] combines federated learning with blockchain for collaborative detection while preserving data confidentiality. MSLSTM [28] leverages wavelet transforms and multiscale LSTM models for anomaly classification. Despite their strong empirical performance, the reliance of these methods on opaque models and their lack of interpretability hinder operational adoption.

In contrast, several rule-based (white-box) approaches use interpretable logic, namely valley-free rules and deterministic inferred AS relationships, to detect route leaks. However, there is no comprehensive and publicly available dataset for AS relationships. Several methods have been proposed to infer AS relationships from BGP topology and routing behaviors [14]–[22]. AS Rank [14] applies valley-free assumptions and topological heuristics. The widely used CAIDA AS-relationships dataset [18] builds upon AS Rank, augmenting it with additional data sources. ProbLink [15] introduces a probabilistic model, TopoScope [16] further refines this with ensemble learning to address observation bias, and HELA [19] extends this into a modular framework. UNARI [20] models deterministic inference uncertainty. This differs from our approach, which assigns a probability distribution to each AS link. ASIRIA [21] infers AS relationships from IRR data, while LeMon [22] validates routes using offline datasets and operator feedback. However, these methods often yield high *FPR* when applied to ASPA-based route leak detection.

Our work complements other BGP security mechanisms, together providing more comprehensive protection against routing anomalies. ROV relies on ROAs [57] to verify that an AS is authorized to originate a given prefix. This mechanism addresses prefix hijacking but remains orthogonal to `PathProb`, which focuses on validating the legitimacy of the `AS_PATH` attribute. BGPsec [58] enhances security by cryptographically signing `AS_PATH` to prevent forgery. In contrast, `PathProb` targets policy violations where routes are propagated authentically yet violate acceptable routing policies. BGP-iSec [5], an extension of BGPsec, introduces additional safeguards against route leaks. Peerlock [59] mitigates leaks via manually configured filters between large network operators, but it lacks scalability and automation. The Only-to-Customer (OTC) attribute [10] allows an AS to proactively restrict route announcements to peers and providers, thereby helping prevent certain types of leaks. Unlike OTC, `PathProb` reactively detects invalid paths received from neighbors. The two approaches are complementary: OTC may help address specific edge cases where `PathProb` could produce false negatives, suggesting potential synergy in joint deployment.

VII. Conclusion

In this paper, we propose `PathProb`, a novel probabilistic framework designed to defend against BGP route leaks—a critical threat to the global routing stability. `PathProb` infers probabilistic distributions for AS links and calculates legitimacy scores for AS paths to determine whether they violate the expected routing propagation policies. `PathProb` not only serves as a supplement to the authoritative declared AS relationships, such as RPKI-based ASPA, but also enhances the detection of route leaks by modeling relationship uncertainty, outperforming deterministic heuristics. It offers operators a flexible threshold adjustment to balance security sensitivity and false alarms, adapting to diverse network environments. Critically, `PathProb` provides immediate, robust protection with minimal overhead while incentivizing a transition to a fully verifiable, white-box routing security ecosystem in the future.

Acknowledgment

We sincerely thank the anonymous reviewers for their insightful and constructive suggestions. This work was supported by the National Key R&D Program of China under Grant No.2022YFB3104800. The corresponding author of this paper is Yanbiao Li.

References

[1] Y. Rekhter, S. Hares, and T. Li, "A Border Gateway Protocol 4 (BGP-4)," RFC 4271, Jan. 2006. [Online]. Available: https://www.rfc-editor.org/info/rfc4271

[2] S. L. Murphy, "BGP Security Vulnerabilities Analysis," RFC 4272, Jan. 2006. [Online]. Available: https://www.rfc-editor.org/info/rfc4272

[3] K. R. B. Butler, T. R. Farley, P. D. McDaniel, and J. Rexford, "A survey of BGP security issues and solutions," *Proc. IEEE*, vol. 98, no. 1, pp. 100–122, 2010. [Online]. Available: https://doi.org/10.1109/JPROC.2009.2034031

[4] G. Huston, M. Rossi, and G. J. Armitage, "Securing BGP - A literature survey," *IEEE Commun. Surv. Tutorials*, vol. 13, no. 2, pp. 199–222, 2011. [Online]. Available: https://doi.org/10.1109/SURV.2011.041010.00041

[5] C. Morris, A. Herzberg, B. Wang, and S. Secondo, "BGP-iSec: Improved security of internet routing against post-rov attacks," in *31st Annual Network and Distributed System Security Symposium, NDSS 2024, San Diego, California, USA, February 26 - March 1, 2024*. The Internet Society, 2024. [Online]. Available: https://www.ndss-symposium.org/ndss-paper/bgp-isec-improved-security-of-internet-routing-against-post-rov-attacks/

[6] Qrator Labs, "Q3 2025 DDoS, bad bots, and BGP incidents statistics and overview." [Online]. Available: https://blog.qrator.net/en/q3-2025-ddos-bad-bots-and-bgp-incidents-statistics_220/

[7] M. Lepinski and S. Kent, "An Infrastructure to Support Secure Internet Routing," RFC 6480, Feb. 2012. [Online]. Available: https://www.rfc-editor.org/info/rfc6480

[8] P. Mohapatra, J. Scudder, D. Ward, R. Bush, and R. Austein, "BGP Prefix Origin Validation," RFC 6811, Jan. 2013. [Online]. Available: https://www.rfc-editor.org/info/rfc6811

[9] J. L. S. Damas, A. Robachevski, L. Blunk, and F. Parent, "Routing Policy Specification Language next generation (RPSLng)," RFC 4012, Mar. 2005. [Online]. Available: https://www.rfc-editor.org/info/rfc4012

[10] A. Azimov, E. Bogomazov, R. Bush, K. Patel, and K. Sriram, "Route Leak Prevention and Detection Using Roles in UPDATE and OPEN Messages," RFC 9234, May 2022. [Online]. Available: https://www.rfc-editor.org/info/rfc9234

[11] B. Todorovic, "Preventing route leaks made simple: BGP roleplay with Junos (RFC 9234)," September 2025. [Online]. Available: https://blog.apnic.net/2025/09/05/preventing-route-leaks-made-simple-bgp-roleplay-with-junos-rfc-9234/

[12] Qrator Labs. [Online]. Available: https://blog.qrator.net/en/reports/

[13] Doug Madory, "BGP routing leak leads to spike of misdirected traffic," 2024. [Online]. Available: https://www.kentik.com/analysis/BGP-Routing-Leak-Leads-to-Spike-of-Misdirected-Traffic

[14] M. J. Luckie, B. Huffaker, A. Dhamdhere, V. Giotsas, and kc claffy, "AS relationships, customer cones, and validation," in *Proceedings of the 2013 Internet Measurement Conference, IMC 2013, Barcelona, Spain, October 23-25, 2013*, K. Papagiannaki, P. K. Gummadi, and C. Partridge, Eds. ACM, 2013, pp. 243–256. [Online]. Available: https://doi.org/10.1145/2504730.2504735

[15] Y. Jin, C. Scott, A. Dhamdhere, V. Giotsas, A. Krishnamurthy, and S. Shenker, "Stable and practical AS relationship inference with ProbLink," in *16th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2019, Boston, MA, February 26-28, 2019*, J. R. Lorch and M. Yu, Eds. USENIX Association, 2019, pp. 581–598. [Online]. Available: https://www.usenix.org/conference/nsdi19/presentation/jin

[16] Z. Jin, X. Shi, Y. Yang, X. Yin, Z. Wang, and J. Wu, "TopoScope: Recover AS relationships from fragmentary observations," in *IMC '20: ACM Internet Measurement Conference, Virtual Event, USA, October 27-29, 2020*. ACM, 2020, pp. 266–280. [Online]. Available: https://doi.org/10.1145/3419394.3423627

[17] G. D. Battista, T. Erlebach, A. Hall, M. Patrignani, M. Pizzonia, and T. Schank, "Computing the types of the relationships between autonomous systems," *IEEE/ACM Trans. Netw.*, vol. 15, no. 2, pp. 267–280, 2007. [Online]. Available: http://doi.acm.org/10.1145/1279660.1279662

[18] CAIDA, "The caida as relationships dataset, 2024 – 2025." [Online]. Available: https://www.caida.org/catalog/datasets/as-relationships/

[19] X. Shi, Z. Jin, B. Xiong, X. Huang, X. Xi, D. Li, H. Zhang, Z. Wang, and X. Yin, "HELA: inferring AS relationships with a hybrid of empirical and learning algorithms," *IEEE Trans. Netw.*, vol. 33, no. 5, pp. 2648–2663, 2025. [Online]. Available: https://doi.org/10.1109/TON.2025.3572148

[20] G. Feng, S. Seshan, and P. Steenkiste, "UNARI: An uncertainty-aware approach to as relationships inference," in *Proceedings of the 15th International Conference on Emerging Networking Experiments And Technologies, CoNEXT 2019, Orlando, FL, USA, December 09-12, 2019*, A. Mohaisen and Z. Zhang, Eds. ACM, 2019, pp. 272–284. [Online]. Available: https://doi.org/10.1145/3359989.3365420

[21] M. Bagnulo, A. García-Martínez, S. Angieri, A. Lutu, and J. Yang, "Practicable route leak detection and protection with ASIRIA," *Comput. Networks*, vol. 211, p. 108966, 2022. [Online]. Available: https://doi.org/10.1016/j.comnet.2022.108966

[22] H. Schulmann and S. Zhao, "Poster: Lemon: Global route leak monitoring service," in *Proceedings of the ACM SIGCOMM 2023 Conference, ACM SIGCOMM 2023, New York, NY, USA, 10-14 September 2023*, H. Schulzrinne, V. Misra, E. Kohler, and D. A. Maltz, Eds. ACM, 2023, pp. 1111–1113. [Online]. Available: https://doi.org/10.1145/3603269.3610852

[23] L. Gao, "On inferring autonomous system relationships in the internet," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 733–745, 2001. [Online]. Available: https://doi.org/10.1109/90.974527

[24] Y. Chen, Q. Yin, Q. Li, Z. Liu, K. Xu, Y. Xu, M. Xu, Z. Liu, and J. Wu, "Learning with semantics: Towards a semantics-aware routing anomaly detection system," in *33rd USENIX Security Symposium, USENIX Security 2024, Philadelphia, PA, USA, August 14-16, 2024*, D. Balzarotti and W. Xu, Eds. USENIX Association, 2024. [Online]. Available: https://www.usenix.org/conference/usenixsecurity24/presentation/chen-yihao

[25] J. Li, J. Cao, Z. Meng, R. Xie, and M. Xu, "RoLL: Real-time and accurate route leak location with AS triplet features," in *IEEE International Conference on Communications, ICC 2023, Rome, Italy, May 28 - June 1, 2023*. IEEE, 2023, pp. 5240–5246. [Online]. Available: https://doi.org/10.1109/ICC45041.2023.10278878

[26] J. Li, J. Cao, Z. Meng, R. Xie, Q. Li, Y. Yang, and M. Xu, "RoLL+: Real-time and accurate route leak locating with AS triplet features at scale," *IEEE/ACM Trans. Netw.*, vol. 32, no. 6, pp. 5263–5278, 2024. [Online]. Available: https://doi.org/10.1109/TNET.2024.3458943

[27] M. Zeng, D. Li, P. Zhang, K. Xie, and X. Huang, "Federated route leak detection in inter-domain routing with privacy guarantee," *ACM Trans. Internet Techn.*, vol. 23, no. 1, pp. 12:1–12:22, 2023. [Online]. Available: https://doi.org/10.1145/3561051

[28] M. Cheng, Q. Li, J. Lv, W. Liu, and J. Wang, "Multi-scale LSTM model for BGP anomaly classification," *IEEE Trans. Serv. Comput.*, vol. 14, no. 3, pp. 765–778, 2021. [Online]. Available: https://doi.org/10.1109/TSC.2018.2824809

[29] Y. Dong, Q. Li, R. O. Sinnott, Y. Jiang, and S. Xia, "ISP self-operated BGP anomaly detection based on weakly supervised learning," in *29th IEEE International Conference on Network Protocols, ICNP 2021, Dallas, TX, USA, November 1-5, 2021*. IEEE, 2021, pp. 1–11. [Online]. Available: https://doi.org/10.1109/ICNP52444.2021.9651957

[30] S. A. E. Monem, A. Khalafallah, and S. I. Shaheen, "BGP route leaks detection using supervised machine learning technique," in *2nd Novel Intelligent and Leading Emerging Sciences Conference, NILES 2020, Giza, Egypt, October 24-26, 2020*. IEEE, 2020, pp. 15–20. [Online]. Available: https://doi.org/10.1109/NILES50944.2020.9257981

[31] G. Asharov, D. Demmler, M. Schapira, T. Schneider, G. Segev, S. Shenker, and M. Zohner, "Privacy-preserving interdomain routing at Internet scale," *Proc. Priv. Enhancing Technol.*, vol. 2017, no. 3, p. 147, 2017. [Online]. Available: https://doi.org/10.1515/popets-2017-0033

[32] S. Kastanakis, "The good, the bad, and the ugly of BGP routing modeling: Confounding factors, selective announcements and location-aware simulations," Ph.D. dissertation, Lancaster University, 2025.

[33] V. Giotsas, M. J. Luckie, B. Huffaker, and K. C. Claffy, "Inferring complex AS relationships," in *Proceedings of the 2014 Internet Measurement Conference, IMC 2014, Vancouver, BC, Canada, November 5-7, 2014*, C. Williamson, A. Akella, and N. Taft, Eds. ACM, 2014, pp. 23–30. [Online]. Available: https://doi.org/10.1145/2663716.2663743

[34] R. Anwar, H. Niaz, D. R. Choffnes, Í. S. Cunha, P. Gill, and E. Katz-Bassett, "Investigating interdomain routing policies in the wild," in *Proceedings of the 2015 ACM Internet Measurement Conference, IMC 2015, Tokyo, Japan, October 28-30, 2015*, K. Cho, K. Fukuda, V. S. Pai, and N. Spring, Eds. ACM, 2015, pp. 71–77. [Online]. Available: https://doi.org/10.1145/2815675.2815712

[35] L. Gao and J. Rexford, "Stable internet routing without global coordination," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 681–692, 2001. [Online]. Available: https://doi.org/10.1109/90.974523

[36] V. Giotsas and S. Zhou, "Valley-free violation in internet routing - analysis based on BGP community data," in *Proceedings of IEEE International Conference on Communications, ICC 2012, Ottawa, ON, Canada, June 10-15, 2012*. IEEE, 2012, pp. 1193–1197. [Online]. Available: https://doi.org/10.1109/ICC.2012.6363987

[37] S. Y. Qiu, P. D. McDaniel, and F. Monrose, "Toward valley-free inter-domain routing," in *Proceedings of IEEE International Conference on Communications, ICC 2007, Glasgow, Scotland, UK, 24-28 June 2007*. IEEE, 2007, pp. 2009–2016. [Online]. Available: https://doi.org/10.1109/ICC.2007.334

[38] A. Azimov, E. Bogomazov, R. Bush, K. Patel, J. Snijders, and K. Sriram, "BGP AS_PATH Verification Based on Autonomous System Provider Authorization (ASPA) Objects," Internet Engineering Task Force, Internet-Draft draft-ietf-sidrops-aspa-verification-22, Mar. 2025, work in Progress. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-sidrops-aspa-verification/22/

[39] rpki-client, n.d. [Online]. Available: https://console.rpki-client.org/aspa.html

[40] Cloudflare Radar. [Online]. Available: https://radar.cloudflare.com/routing#routing-anomalies

[41] L. Subramanian, S. Agarwal, J. Rexford, and R. H. Katz, "Characterizing the internet hierarchy from multiple vantage points," in *Proceedings IEEE INFOCOM 2002, The 21st Annual Joint Conference of the IEEE Computer and Communications Societies, New York, USA, June 23-27, 2002*. IEEE Computer Society, 2002, pp. 618–627. [Online]. Available: https://doi.org/10.1109/INFCOM.2002.1019307

[42] K. Sriram, D. Montgomery, D. R. McPherson, E. Osterweil, and B. Dickson, "Problem Definition and Classification of BGP Route Leaks," RFC 7908, Jun. 2016. [Online]. Available: https://www.rfc-editor.org/info/rfc7908

[43] Gurobi Optimization, LLC, 2024. [Online]. Available: https://www.gurobi.com

[44] T. Achterberg, "SCIP: solving constraint integer programs," *Mathematical Programming Computation*, vol. 1, no. 1, pp. 1–41, Jul 2009. [Online]. Available: https://doi.org/10.1007/s12532-008-0001-1

[45] S. Kastanakis, V. Giotsas, I. Livadariu, and N. Suri, "Replication: 20 years of inferring interdomain routing policies," in *Proceedings of the 2023 ACM on Internet Measurement Conference, IMC 2023, Montreal, QC, Canada, October 24-26, 2023*, M. Montpetit, A. Leivadeas, S. Uhlig, and M. Javed, Eds. ACM, 2023, pp. 16–29. [Online]. Available: https://doi.org/10.1145/3618257.3624799

[46] G. Huston. (2025) BGP updates in 2024. [Online]. Available: https://blog.apnic.net/2025/01/07/bgp-updates-in-2024/

[47] T. Green, A. Lambert, C. Pelsser, and D. Rossi, "Leveraging inter-domain stability for BGP dynamics analysis," in *Passive and Active Measurement*, R. Beverly, G. Smaragdakis, and A. Feldmann, Eds. Cham: Springer International Publishing, 2018, pp. 203–215.

[48] G. Comarela, G. Gürsun, and M. Crovella, "Studying interdomain routing over long timescales," in *Proceedings of the 2013 Internet Measurement Conference, IMC 2013, Barcelona, Spain, October 23-25,*

2013, K. Papagiannaki, P. K. Gummadi, and C. Partridge, Eds. ACM, 2013, pp. 227–234. [Online]. Available: https://doi.org/10.1145/2504730.2504771

[49] K. R. B. Butler, P. D. McDaniel, and W. Aiello, "Optimizing BGP security by exploiting path stability," in *Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, Alexandria, VA, USA, October 30 - November 3, 2006*, A. Juels, R. N. Wright, and S. D. C. di Vimercati, Eds. ACM, 2006, pp. 298–310. [Online]. Available: https://doi.org/10.1145/1180405.1180442

[50] Route Views, "University of Oregon Route Views Project," n.d. [Online]. Available: http://www.routeviews.org/routeviews/

[51] RIS, "RIPE (RIS)," n.d. [Online]. Available: https://www.ripe.net/ris/

[52] Wikipedia contributors, "Tier 1 network — Wikipedia, the free encyclopedia." [Online]. Available: https://en.wikipedia.org/wiki/Tier_1_network

[53] PeeringDB, n.d. [Online]. Available: https://www.peeringdb.com

[54] CAIDA, "The CAIDA UCSD AS to organization mapping dataset," 2014. [Online]. Available: https://www.caida.org/catalog/datasets/as-organizations/

[55] J. Furuness, C. Morris, R. Morillo, A. Kasiliya, B. Wang, and A. Herzberg, "Securing BGP ASAP: ASPA and other post-rov defenses," in *32nd Annual Network and Distributed System Security Symposium, NDSS 2025, San Diego, California, USA, February 24-28, 2025*. The Internet Society, 2025. [Online]. Available: https://www.ndss-symposium.org/ndss-paper/securing-bgp-asap-aspa-and-other-post-rov-defenses/

[56] J. Furuness, C. Morris, R. Morillo, A. Herzberg, and B. Wang, "Bgpy: The BGP python security simulator," in *2023 Cyber Security Experimentation and Test Workshop, CSET 2023, Marina del Rey, CA, USA, August 7-8, 2023*. ACM, 2023, pp. 41–56. [Online]. Available: https://doi.org/10.1145/3607505.3607509

[57] J. Snijders, B. Maddison, M. Lepinski, D. Kong, and S. Kent, "A Profile for Route Origin Authorizations (ROAs)," RFC 9582, May 2024. [Online]. Available: https://www.rfc-editor.org/info/rfc9582

[58] M. Lepinski and K. Sriram, "BGPsec Protocol Specification," RFC 8205, Sep. 2017. [Online]. Available: https://www.rfc-editor.org/info/rfc8205

[59] T. McDaniel, J. M. Smith, and M. Schuchard, "Flexsealing BGP against route leaks: peerlock active measurement and analysis," *arXiv preprint arXiv:2006.06576*, 2020.

Appendix A
Triple-Minimum vs Full-Path Scoring for Long AS Paths

We compare the effectiveness of the triple-minimum score (Eq. (1)) versus the full-path score (Eq. (2)) for handling long AS paths. Focusing on paths in the `Update_Event_VD` dataset with length ≥ 7, we plot the legitimacy score distributions for legitimate and leaked paths in Fig. 13 and Fig. 14, respectively.

The comparison highlights the advantage of the triple-minimum strategy. Although score decay along long paths is an inherent limitation of the probabilistic model, the triple-minimum score substantially mitigates this impact. In Fig. 13, legitimate paths maintain high scores, rarely falling below 0.2, and exhibit only minor overlap with leaked paths around 0.3. In contrast, the full-path score (Fig. 14) experiences rapid decay, leading to significant overlap between legitimate and leaked paths in the $[0, 0.1]$ range. These findings validate the theoretical analysis in Section III-B: by limiting the scope of probability multiplication, the triple-minimum scheme mitigates the excessive accumulation of length-induced uncertainty. This ensures that the metric maintains strong discriminability between legitimate and leaked routes, even for long AS paths.

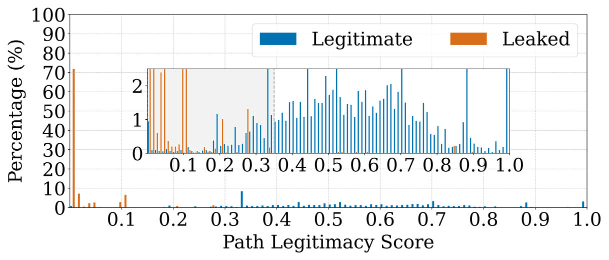


Fig. 13. Triple-minimum score distribution of legitimate and leaked paths.

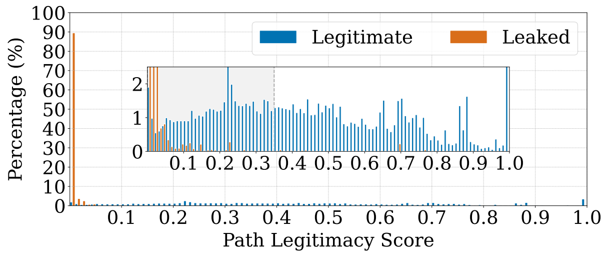


Fig. 14. Full-path score distribution of legitimate and leaked paths.

Appendix B
Algorithmic Parameters Sensitivity

We conduct a sensitivity analysis of `PathProb` for two key parameters: the number of Gibbs samples (K) in Section IV-D and the core-to-edge threshold (τ) in Section IV-E. For each configuration, we measure AS relationship inference accuracy using `CAIDA` and `ASPA` datasets and route leak detection performance—measured by average precision, recall, and FPR across all three dimensions—with results summarized in Tables IV and V.

A. The Number of Gibbs Samples

Varying the number of Gibbs samples K from 500 to 4000 (with fixed $\tau = 0.8$) demonstrates that `PathProb` exhibits high stability once K reaches a sufficient magnitude. As shown in Table IV, the variation in AS relationship accuracy is negligible (< 0.3 percentage points) across both datasets. Route leak detection metrics exhibit similar stability: recall remains constant at 98.39%, while precision fluctuates marginally ($95.43\% \sim 95.64\%$) and FPR exhibits a slight improvement from 4.75% to 4.52%. Given this flat convergence region, we select $K = 1000$ as the default setting to achieve an optimal balance between detection performance and computational efficiency.

B. Core-to-edge threshold

Table V analyzes the sensitivity of `PathProb` to the core-to-edge threshold (τ) over the range [0.6, 0.99]. Increasing τ monotonically improves AS relationship inference accuracy on the `CAIDA` dataset (from 93.31% to 97.42%), but for $\tau > 0.8$, it leads to a gradual decline in AS relationship reference accuracy on `ASPA` dataset, with accuracy dropping from 93.93% to 91.91%. For route leak detection, moderate thresholds ($\tau \in [0.6, 0.8]$) maintain high precision and stable

TABLE IV
Sensitivity to the Number of Gibbs Samples.

K	AS rel. acc. (%)		route leak avg. (%)		
	CAIDA	ASPA	Precision	Recall	FPR
500	95.92%	93.93%	95.59%	98.39%	4.57%
750	95.70%	93.93%	95.48%	98.39%	4.69%
1000	**95.94%**	**93.93%**	**95.43%**	**98.39%**	**4.75%**
2000	95.67%	93.64%	95.63%	98.39%	4.53%
4000	95.69%	93.64%	95.64%	98.39%	4.52%

TABLE V
Sensitivity to the core-to-edge threshold.

τ	AS rel. acc. (%)		route leak avg. (%)		
	CAIDA	ASPA	Precision	Recall	FPR
0.6	93.31%	93.93%	95.71%	98.39%	4.44%
0.7	94.44%	93.93%	95.70%	98.39%	4.45%
0.8	**95.94%**	**93.93%**	**95.43%**	**98.39%**	**4.75%**
0.9	96.51%	93.35%	94.13%	98.78%	6.21%
0.99	97.42%	91.91%	90.60%	98.89%	10.43%

TABLE VI
More information about `AS_Rel_VD` validation dataset.

	time YYYY/MM	# of links	# of ASes
`ASPA` Dataset	2025/07	557	303
`CAIDA` Dataset	2024/08	571,873	77,242
	2024/09	574,352	77,246
	2024/10	698,865	77,349
	2024/11	700,774	77,344
	2024/12	579,026	77,500
	2025/01	689,189	77,480
	2025/02	699,461	77,580
	2025/03	709,737	77,600
	2025/04	582,792	77,728
	2025/05	569,254	77,792
	2025/06	571,399	78,144
	2025/07	554,874	78,026

recall. Conversely, overly strict thresholds ($\tau \geq 0.9$) yield only marginal recall gains (up to 98.89%) but severely compromise precision (dropping to 90.60%) and increase the FPR sharply to 10.43%. Consequently, we configure $\tau = 0.8$ as the default setting. This value represents a robust operating point that strikes an optimal balance: ensuring high accuracy across both `CAIDA` and `ASPA` datasets while minimizing false alarms.

Appendix C
More Details about Datasets

This appendix provides detailed specifications of the datasets used in Section V.

Table VII presents a detailed statistical summary of the two BGP RIB datasets, `RIB_Year` and `RIB_Day`. For each dataset, we list the total number of AS paths, the number of unique paths, the number of unique AS links inferred from these paths, and the total number of unique ASes.

`RIB_Year` consists of 12 sets, each corresponding to a month from August 2024 to July 2025, and constructed by aggregating the RIBs collected during the first five days of the respective month.

`RIB_Day` contains 30 RIB sets from July 1st to July 10th, 2025. Every three sets correspond to a single day, with one set representing an 8-hour interval within that day.

Table VI provides detailed information of the `AS_Rel_VD` validation dataset for evaluating AS relationship inference schemes. It consists of two parts: the high-confidence but limited-coverage `ASPA` dataset and the broad-coverage but lower-confidence `CAIDA` dataset.

Table VIII provides a statistics overview of the `Update_Event_VD`, constructed based on Cloudflare Radar-reported route leak events from June 16 to June 30, 2025. To verify the detection robustness of `PathProb` from multiple dimensions, we partition the dataset into finer-grained categories based on time, collectors, and network roles.

TABLE VII
More information about RIB_Year and RIB_Day AS Path Datasets.

	time YYYY/MM/ DD.HH	# of total paths	# of unique paths	# of unique links	# of unique ASes	time YYYY/MM/ DD.HH	# of total paths	# of unique paths	# of unique links	# of unique ASes
RIB_Year Dataset	2024/08	17,812,787,966	50,097,329	538,618	84,268	2025/02	18,044,031,402	49,423,726	538,555	84,252
	2024/09	17,780,336,699	50,124,183	538,274	84,140	2025/03	17,745,253,614	48,413,752	535,735	84,262
	2024/10	17,402,694,208	49,089,812	536,714	84,160	2025/04	17,770,669,084	49,844,198	542,200	84,256
	2024/11	17,587,402,826	48,590,671	536,735	84,099	2025/05	17,616,132,681	48,570,732	536,409	84,269
	2024/12	16,730,401,202	49,278,366	549,913	84,232	2025/06	17,737,310,048	50,390,243	536,283	84,447
	2025/01	16,459,646,891	46,176,443	530,784	84,148	2025/07	17,688,500,604	50,764,884	539,820	84,340
RIB_Day Dataset	2025/07/01.00	562,245,893	45,476,796	527,869	84,141	2025/07/06.00	561,328,617	45,054,202	520,897	84,174
	2025/07/01.08	559,912,216	45,296,390	527,175	84,150	2025/07/06.08	562,539,375	45,142,241	521,273	84,171
	2025/07/01.16	562,285,152	45,571,259	526,300	84,145	2025/07/06.16	562,559,232	45,164,465	521,326	84,174
	2025/07/02.00	562,309,433	45,544,999	527,234	84,131	2025/07/07.00	561,377,025	45,060,718	518,980	84,164
	2025/07/02.08	562,301,724	45,558,121	527,384	84,163	2025/07/07.08	561,073,243	45,031,075	520,653	84,188
	2025/07/02.16	560,867,760	45,206,765	520,748	84,163	2025/07/07.16	560,290,282	45,190,192	519,763	84,190
	2025/07/03.00	561,220,928	45,092,003	520,863	84,145	2025/07/08.00	559,203,196	45,034,167	517,492	84,183
	2025/07/03.08	560,683,214	45,086,457	521,116	84,166	2025/07/08.08	557,798,341	45,082,550	519,198	84,204
	2025/07/03.16	562,071,426	45,124,436	520,702	84,189	2025/07/08.16	559,816,349	45,202,958	521,023	84,225
	2025/07/04.00	562,458,386	45,198,333	521,061	84,168	2025/07/09.00	560,886,180	45,324,225	521,116	84,212
	2025/07/04.08	562,633,120	45,200,883	521,562	84,180	2025/07/09.08	561,307,680	45,286,751	521,120	84,237
	2025/07/04.16	562,589,209	45,243,791	521,531	84,185	2025/07/09.16	561,480,507	45,254,241	521,610	84,251
	2025/07/05.00	562,524,870	45,187,424	520,222	84,177	2025/07/10.00	561,534,575	45,283,230	521,896	84,250
	2025/07/05.08	562,396,680	45,189,392	519,988	84,173	2025/07/10.08	560,494,811	45,129,055	519,731	84,256
	2025/07/05.16	562,531,883	45,161,474	521,356	84,174	2025/07/10.16	561,829,075	45,236,905	521,864	84,263

TABLE VIII
Statistics of Legitimated and Leaked Paths in Update_Event_VD across Time, Collectors, and AS Roles.

Dimension	Instance	# of legitimate path	# of leaked path	Instance	# of legitimate path	# of leaked path	Instance	# of legitimate path	# of leaked path
Time	2025/06/16	1,407,170,355	457,003	2025/06/21	1,315,961,858	1,036,910	2025/06/26	1,496,280,402	221,872
	2025/06/17	1,502,800,928	505,884	2025/06/22	1,319,472,109	164,468	2025/06/27	1,477,188,830	184,128
	2025/06/18	1,497,639,478	627,374	2025/06/23	1,529,775,235	432,016	2025/06/28	1,295,223,655	158,630
	2025/06/19	1,559,610,899	425,391	2025/06/24	1,464,021,082	280,344	2025/06/29	1,324,506,360	155,173
	2025/06/20	1,527,752,163	486,090	2025/06/25	1,410,563,199	180,324	2025/06/30	1,455,365,959	277,698
Collector	rrc00	2,911,494,005	454,423	rrc14	266,872,872	53,922	napafrica	1,347,195,471	244,377
	rrc01	1,165,374,129	385,474	rrc15	745,810,579	327,291	nwax	6,809,808	1,765
	rrc03	1,422,240,840	440,226	rrc16	114,494,423	34,387	perth	325,032,836	91,935
	rrc04	209,308,682	47,660	rrc18	81,023,634	6,847	route-views2	268,855,033	76,808
	rrc05	1,098,933,245	280,067	rrc19	768,821,357	108,373	route-views3	622,032,405	127,316
	rrc06	120,403,504	46,055	rrc20	2,439,433,520	347,526	sfmix	413,325,454	59,108
	rrc07	346,834,438	145,609	rrc21	669,131,270	242,231	sg	674,367,418	214,802
	rrc10	378,292,894	106,240	eqix	559,750,399	164,746	soxrs	34,344,141	78,086
	rrc11	304,515,412	90,114	isc	249,791,394	58,984	sydney	512,035,425	129,384
	rrc12	1,571,538,321	450,448	kixp	362,942,821	47,779	telxatl	164,495,001	41,701
	rrc13	257,140,170	66,519	linx	1,117,890,129	601,348	wide	52,801,482	21,754
Role	T1	8,963,194,571	4,210,093	Cont	4,831,447,690	1,746,464	PT	86,409,130	8,911
	Stub	3,240,136,429	362,469	HI	15,698,651,515	4,752,318	Hyb	605,983,342	853,890
	Trans	21,580,977,672	5,593,305	Sib	2,063,882,891	273,857			

Appendix D
Artifact Appendix

We put forward a new paradigm for BGP route-leak detection. This is achieved by: (i) inferring probabilistic AS business relationships for each AS link from AS paths extracted from global BGP routing data, and (ii) leveraging these probabilities to calculate a legitimate score for each path, thereby facilitating the detection of route leaks.

To facilitate rapid validation of our core methodology, the supplied experiments are streamlined, scaled-down versions of the full-scale studies detailed in the paper.

A. Description & Requirements

1) How to access: Users can access our artifact in our public repository at https://doi.org/10.5281/zenodo.17920056 and https://github.com/hyq8868/PathProb

2) Hardware dependencies: : x86-64 CPU ($\geq$8 cores), $\geq$16 GB RAM, and $\geq$10 GB free disk space.

3) Software dependencies:

- Operating System: Ubuntu 20.04 LTS (tested by the authors).
- System Packages (Ubuntu `apt-get`): `git`, `wget`, `zstd`, `build-essential`, `python3-pip`, `python3-venv`, `graphviz`, `libjpeg-dev`, and `zlib1g-dev`.
- Runtimes: Python 3.8+ and PyPy 3.10+.
  Note: The default PyPy package on Ubuntu 20.04 or 22.04 may be an older version. For experiment E3, please manually install PyPy 3.10+. An example installation procedure is provided below.

```
wget https://downloads.python.org/pypy/pypy3.10-
    v7.3.16-linux64.tar.bz2
tar -xjf pypy3.10-v7.3.16-linux64.tar.bz2
export PATH=$(pwd)/pypy3.10-v7.3.16-linux64/bin:
    $PATH
pypy3 -m ensurepip
pypy3 --version # Verify
```

- Dependencies: All required packages are one-command installable via pip using the supplied `requirements.txt` and `requirements_pypy.txt`.

4) Benchmarks: The datasets required for the experiments are provided as follows:

- Ready-to-use data: pre-processed and saved in `test_data/`. *Note*: The datasets below are scaled-down subsets. To reproduce the full-scale experiments, please use the download script provided in our GitHub repository.
  - `Rib_Year`: A scaled-down AS path dataset collected from June 2025, used as input for probabilistic AS relationship inference (E1 in Section D-E1). Located at: `test_data/prob_inference/paths/202506/`
  - `AS_Rel_VD`: ASPA objects and CAIDA dataset, provide the ground truth for validating the relationship inference results (E1 in Section D-E1). Located at: `test_data/prob_inference/validation/`
  - `Update_Event_VD`: A labeled collection of legitimate and leaked AS paths serves as the ground-truth for validating the route-leak detector (E2 in Section D-E2). Located at: `test_data/leak_detection/cloudflare_data/`, with network roles in `as_role`.
- Auto-download data: automatically fetched and decompressed.
  - `CAIDA Serial-1`: This dataset is used by the simulator to build the topology for Experiment E3 in Section D-E3. It will be downloaded automatically by the simulator into the directory: `pathprob_sim/data/cache/`

B. Artifact Installation & Configuration

```
# Install system packages
sudo apt-get update
sudo apt-get install -y git build-essential graphviz
     libjpeg-dev zlib1g-dev wget zstd python3-pip
    python3-venv

# Change into the directory
cd PathProb

# Prepare a Python3 virtual environment; install
    dependencies
python3 -m venv .python_venv
source .python_venv/bin/activate
python3 -m pip install --upgrade pip
python3 -m pip install --upgrade wheel
pip install -r requirements.txt
deactivate

# Prepare PyPy virtual environment and its
    dependencies
pypy3 -m venv .pypy_venv
source .pypy_venv/bin/activate
pypy3 -m pip install pip --upgrade
pypy3 -m pip install wheel --upgrade
pypy3 -m pip install -r requirements_pypy.txt
deactivate

# Extract ready-to-use data
zstd -d test_data.tar.zst -c | tar -xf -
```

C. Experiment Workflow

The experimental workflow is divided into three sequential phases.

1) **Probabilistic AS-relationship inference** - using the June-2025 AS-path dataset (`RIB_Year`), we infer business relationship probabilities and evaluate their accuracy against the CAIDA / ASPA ground-truth (`AS_Rel_VD`).
2) **Route-leak detection** - fed with the above probabilistic AS-relationships, we compute the legitimacy score for each AS path to identify route leaks, and evaluate `PathProb`'s *precision*, *recall*, and *FPR* (False Positive Rate) against dataset `Update_Event_VD`.
3) **Large-scale simulation** - the simulator auto-downloads the AS relationships archived by `CAIDA` to construct the global topology, injects synthetic leaks and outputs LIR (Leakage Infection Rate) and LCR (Legitimate Connection Rate) plots.

D. Major Claims

- (C1) PathProb infers probabilistic AS relationships from AS path inputs and validates them against ASPA objects and the CAIDA Serial-2 dataset. This claim is supported by experiment (E1), which produces representative results aligning with Fig. 6.
- (C2) PathProb computes a legitimacy score for each path to perform route leak detection and achieves the expected performance metrics (*precision*, *recall*, and *FPR*). This claim is supported by experiment (E2), with results aligning with Fig. 9 and Fig. 10.
- (C3) PathProb achieves lower LIR and higher LCR compared to the ASPA scheme through a large-scale simulation on a global AS-level topology. This claim is supported by experiment (E3), with results aligning with Fig. 11 and Fig. 12.

E. Evaluation

1) Experiment (E1): [Probabilistic AS relationship inference] [5 human-minutes + 1 compute-hours]: This experiment infers probabilistic AS relationships from the provided AS-path dataset and validates them against ASPA objects and CAIDA Serial-2 (`AS_Rel_VD`). The expected outcome is a three-way probability distribution (customer–provider, peer–peer, provider-customer) across every AS link in the dataset. The accuracy of these distributions is then evaluated, and the results are aligned with Fig. 6.

Reproducibility Notes:

- Solver: The paper used Gurobi, which requires an academic license, as the ILP solver. To enable reproducibility, this artifact switches to the open-source solver SCIP.
- Dataset: To facilitate rapid validation, this artifact provides a scaled-down subset of the paper's `Rib_Year` dataset, containing only records from June 2025.

The solver and reduction in scale of the dataset does not alter the core characteristics underpinning the qualitative conclusions.

[Preparation] Activate the Python virtual environment.

```
source .python_venv/bin/activate
```

[Execution] Run the inference script.

```
python3 infer_prob/asrel_prob.py \
  --path_dir test_data/prob_inference/paths/202506 \
  --print_dir test_data/prob_inference/result/202506
```

[Results] Note on interpreting results: The paper's Fig. 6 shows aggregate metrics derived from the full-scale experiment. Due to the scaled-down dataset used in this artifact, this script outputs the specific accuracy values corresponding to this single run.

```
python3 eval_asrel.py --probs test_data/
    prob_inference/result/202506/pathprob.txt
```

A similar output as follows should be shown:

```
Results for ASPA validation: accuracy: 94.20%
Results for CAIDA validation: accuracy: 95.79%
```

2) Experiment (E2): [Route-leak detection] [5 human-minutes + 10 compute-minutes]: This experiment uses the probabilistic AS relationships inferred from (E1) to score each AS path and perform route-leak detection. The expected outcome is a set of *Precision*, *Recall*, and *FPR* metrics, and the results are aligned with Fig. 9 and Fig. 10.

Reproducibility Notes: To ensure rapid execution, this artifact provides a scaled-down subset of the paper's `UPDATE_EVENT_VD` dataset, containing only records of five days, namely days 16-20 of June 2025.

[Preparation] Activate the Python virtual environment.

```
source .python_venv/bin/activate
```

[Execution] Run the detection script. This script saves the detailed metrics to `test_data/leak_detection/result/`, and also prints the summary metrics directly to the console.

```
python3 route_leak_detection.py
```

A similar output as follows should be shown:

```
------------------------------------------
Category Precision Recall FPR
-----------------------------------------
time 96.85% 98.82% 3.22%
collector 96.80% 98.91% 3.36%
role 96.59% 96.71% 3.41%
----------------------------------------
Details are saved to test_data/leak_detection/result
```

[Results] Generate the plots from the saved metrics. Results are saved to the directory `test_data/leak_detection/result/`.

```
python3 figure_routeleak.py
deactivate
```

3) Experiment (E3): [Simulation] [5 human-minutes + 3 compute-hours]: This experiment runs a large-scale simulation to measure *LIR* and *LCR* across various deployment scenarios. The expected outcome is a set of plots demonstrating that PathProb achieves a lower *LIR* and a higher *LCR* than the ASPA scheme, and the results are aligned with Fig. 11 and Fig. 12.

Reproducibility Notes: To reduce the overall execution time, for each deployment rate of {25%, 50%, 75%, 100%}, we use ASPA's issuance rates of {0%, 25%, 50%, 75%, 100%} with 100 trials per configuration.

[Preparation] This simulation requires `PyPy3`. Activate the PyPy virtual environment.

```
source .pypy_venv/bin/activate
```

[Execution] Run the simulation script.

```
export PYTHONHASHSEED=0 # required by simulator BGPy
pypy3 -m pathprob_sim --trials 100
```

[Results] Across all deployment scenarios with different deployment rates and ASPA issuance rates, the script computes LIR and LCR for every evaluated scheme and automatically renders the comparison plots into `pathprob_sim/data/graphs`.

```
pypy3 pathprob_sim/graph/graph.py
```