ROV-MI: Large-Scale, Accurate and Efficient Measurement of ROV Deployment

Wenqi Chen*, Zhiliang Wang*, Dongqi Han*, Chenxin Duan*, Xia Yin†, Jiahai Yang*, Xingang Shi*
*Institute for Network Sciences and Cyberspace, BNRist, Tsinghua University, Beijing, China
†Department of Computer Science and Technology, BNRist, Tsinghua University, Beijing, China
{chenwq19, handq19, dcx19}@mails.tsinghua.edu.cn, {wzl, yang, shixg}@cernet.edu.cn, yxia@tsinghua.edu.cn

***Abstract*—Securing inter-domain routing systems of the Internet from illegitimate prefix annoucements has been a great concern for the researchers and network operators. After the failure of many BGP (Border Gateway Protocol) security enhancement mechanisms to achieve extensive deployment, it is encouraging to see that the deployment of RPKI (Resource Public Key Infrastructure) is gradually increasing worldwide. For a deeper understanding of the impact of RPKI, many studies have been devoted to measuring the deployment of RPKI, including the deployment of ROA (Route Origin Authorization) and ROV (Route Origin Validation). Unlike the measurement of ROA deployment which can be directly derived from the data in RPKI repository, the measurement of ROV deployment requires more sophisticated measurement and inference techniques. However, existing work has limited measurement range, and the inference methods are either inaccurate or inefficient.**

In this paper, we propose a new framework, ROV-MI, for the measurement of ROV deployment, which consist of a large-scale measurement infrastructure driven by in-the-wild RPKI invalid prefixes in the control plane to detect the filtering of these invalid updates with active probing in the data plane, and an efficient and accurate inference algorithm based on Bayesian inference techniques. We implement ROV-MI for measuring real-world ROV deployment and compare it to prior works, and the results show that ROV-MI can accurately infer ROV adoption of $\sim$10 times more ASes (Autonomous Systems) with less than 20% of the execution time compared to current state-of-the-art methods.

I. Introduction

Border Gateway Protocol (BGP) serves as the basis for inter-domain routing systems of the Internet [1], which computes the routes between tens of thousands of smaller networks called Autonomous Systems (ASes). However, BGP lacks mechanisms for route authentication: BGP does not check whether the received routes are valid or not. Thus, an AS can ***hijack*** prefixes it does not control by intentionally or unintentionally announcing invalid routes, making BGP vulnerable to malicious ASes and configuration errors. These vulnerablities may significantly impair the reachability of the Internet or be exploited by attackers to launch man-in-the-middle attacks [2], as is evidenced by some well-known route leak and prefix hijack events [3], [4].

To address this issue, many security enhancement mechanisms for BGP have been proposed and standradized [5]–[7], but most of them have gained limited adoption because of the difficulty of deployment or the lack of incentives (e.g. low benefit when partially deployed). So for a long time, the route authentication for BGP remains unsolved. However, many recent studies and reports show that one of the the mechanisms, RPKI (Resource Public Key Infrastructure) [8], has shown an increasing trend of deployment [9], [10]. RPKI consists of two processes: ROA (Route Origin Authorization) and ROV (Route Origin Validtion). ROA specifies which prefixes each AS can advertise with Digital Certificates as technical basis and Regional Internet Registries (RIRs) as trust anchors, and stores these records in a trustworthy centralized database named RPKI repository. Then the ASes adopting ROV can periodically retrieve ROA records from RPKI repository, and when receiving new route annoucements, check their validity and filter the invalid ones. Recent reports show a promising trend in RPKI deployment, with over 30% of prefixes all over the Internet registered in PRKI by April, 2021 [10], [11] .

To better understand the impact and future benefit of RPKI, many studies have been devoting to measuring RPKI deployment [9], [10], [12]–[15], which consists of ROA and ROV deployment. ROA deployment measurement takes relatively little effort to conduct, as it can fetch data from RPKI repository, and derive most results directly (e.g. how many prefixes/anouncements can be validated by existing ROA records) [10], [12]. Yet compared to the deployment of ROA, we have relatively little understanding of ROV deployment. The main reason is that there is no available database of ROV deployment like RPKI repository, thus requiring more sophisticated measurement and inference methods [9].

In recent years, several studies have attempted to take a step forward in the measurement and inference of ROV deployment [9], [13], [14], which are shown in Table I. Galid *et al.* [9] is one of the earliest efforts, which collects BGP route announcements from public route collectors (e.g. Routeviews, RIPE RIS) to see which paths are filtering

Network and Distributed Systems Security (NDSS) Symposium 2022
24-28 April 2022, San Diego, CA, USA
ISBN 1-891562-74-6
https://dx.doi.org/10.14722/ndss.2022.23214
www.ndss-symposium.org

TABLE I. Comparison of representative ROV measurements.

Method	Measurement	Inference	
	Scale	Accurate	Efficient
Galid *et al.* [9]	Medium	✗	✔
Reuteret *et al.* [13]	Narrow	✔	✔
Gray *et al.* [14]	Medium	✔	✗
ROV-MI	Large-scale	✔	✔

invalid routes, and uses a heuristic to find the ASes that are most likely to adopt ROV from the labeled paths. However, this algorithm can be easily affected by other kinds of filtering (e.g. traffic engineering) and is highly dependent on the distribution of the vantage points [13], thus having relatively low accuracy. To this end, Reuter *et al.* [13] proposes a controlled measurement method that uses PEERING testbed [16] to announce a pair of valid/invalid prefixes to the ASes which peer with it. By comparing whether the prefix pair is accepted, it can accurately measurement the ROV adoption of its peering ASes. The main problem is the connected-restriction imposed on the measured ASes that they have to be directly connected to PEERING testbed, which greatly narrows down the measurement range. Gray *et al.* [14] removes this restriction by extending the method of [13] to collecting the route announcements of the prefix pair advertised with PEERING from public route collectors to label which paths are propagating the invalid prefix and which are filtering them. It then models the problem with Bayesian methods and solve it by MCMC (Markov Chain Monte Carlo) methods. However, the measurement range is still limited because PEERING can only announce prefixes from several certain sites, which means the number of paths observed at public route collectors is limited. What's more, MCMC is a sampling-based method, which is extremely time-consuming and becomes unscalable when the problems is larger, and is hard to access convergence [17].

Our work. In lights of the problems above, we propose ROV-MI, a large-scale, accurate and efficient framework for ROV measurement. In the measurement part, to expand the range of the measurement, we find it feasible to utilize the invalid prefixes in the wild to increase the number of origins that announce invalid updates, and propose to observe the propagation of these updates with active probing, through which we observe $\sim$10 times more paths compared to existing work. In the inference part, we borrow the idea of solving it in a probabilistic setting from [14], but introduce a more superior Bayesian Inference technique, Stein Variational Gradient Descent (SVGD) [18], to solve it. Unlike MCMC, SVGD uses deterministic method to approximate target probability distributions, which can infer the deployment of ROV in an accurate and efficient way.

Contribution. In a nutshell, our contribution can be summarized as follows:

- Proposing a large-scale measurement scheme for ROV deployment. By leveraging the invalid prefixes in the wild, our measurement method greatly enlarges the number of labeled paths, which provides the opportunity for a large-scale ROV deployment measurement.
- Solving the inference of ROV deployment both accurately and efficiently. To ensure the accuracy of the inference, we extend the probabilistic model from previous studies, but the large-scale measurement brings the problem to extremely high-dimensional space. To this end, we introduce a superior variational inference algorithm, SVGD, for an accurate and efficient solution.
- We implement our measurement framework, ROV-MI, which combine the large-scale measurement scheme and the accurate and efficient inference algorithm, to measure the real-world ROV deployment. The results show ROV-MI can accurately measure the ROV deployment in the Internet, outperform existing work in accuracy and efficiency, and can generalize to other problems.

II. Background

This section first introduces how BGP works (§ II-A), then analyzes one major security issue of BGP and introduces how RPKI solves it (§ II-B). Finally we elaborate existing studies about RPKI measurement (§ II-C).

A. Border Gateway Protocol (BGP)

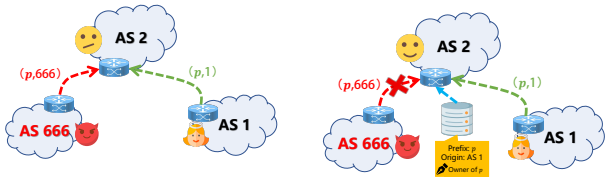
The Internet is composed of many Autonomous Systems (ASes), which are sets of connected Internet Protocol (IP) prefixes, each under the control of some network operators on behalf of a administrative entity (e.g. Internet Service Provider (ISP), business enterprise). Each AS has a AS Number (ASN) and some IP prefixes (i.e. IP address blocks) allocated to it. The ASN uniquely identify its network when exchanging information with other ASes. The prefixes contain many IP addresses, which it can adopt for different usage (e.g. Cloud Services, Enterprise Networks).

Border Gateway Protocol (BGP) is the de facto routing protocol between these ASes, with which each AS can advertise information about its own prefixes and accept information about the path to its external prefixes. The main process of BGP can be summarized as following steps: Fisrt, ASes send UPDATE messages (updates) to peers (i.e. neighbor ASes) to propagate information about its own prefixes. For example, if an AS A owns prefix p, it may send a update containing a ternary (A, p, A) to its peers. The first two terms in the ternary are the ASN and the prefix respectively, and the third term is the path to the prefix p. When other ASes receive the updates, there might be multiple updates containing the same prefix p, and the receiver chooses one from them as the path to the prefix, and then further propagate the prefix to its peers. For example, if an AS E receives two updates $(A, p, C \rightarrow B \rightarrow A)$ and $(A, p, D \rightarrow A)$, which means there are two paths $C \rightarrow B \rightarrow A$ and $D \rightarrow A$ to the prefix p, it may choose $D \rightarrow A$ as the path to p for it is shorter (has fewer hops). Then it prepends its own ASN to the path, and send the update $(A, p, E \rightarrow D \rightarrow A)$ to it peers. In practice

the choice of the path may not only depend on the hops of the path, but complies to the routing policy of the AS which incorporates more complex decision factors (e.g. the commercial relationship of the ASes) [19].

B. RPKI

Although BGP has been serving as the basis of the Internet for a long history, many security issues of it remain unsolved yet, and one of them is the lack of authentication mechanism. BGP assumes no adversaries among the ASes, so it fully trusts all the updates it receives. However, this leaves BGP vulnerable to accidental or malicious disruption: If an AS advertises an update containing a prefix that is not assigned to it (may be caused by adversary ASes or configuration errors), the receivers lack proper methods to check the validity of the update, and may choose the invalid path. For instance, as Fig. 1(a) shows, AS 666 may anounce a prefix p which belongs to AS 1, and when AS 2 receives both the valid and invalid updates, it cannot tell which one is the valid. This behavior, also known as BGP Hijacking, may result in global reachability failures and economic losses [2], as is evidented by many famous accidents like Pakistan Telecom's hijacking of Youtube traffic in Feb, 2008 [3].



(a) Original BGP. (b) BGP with RPKI deployed.

Fig. 1: The propagation of valid and invalid updates before/after RPKI is deployed. RPKI helps ASes with detecting and filtering invalid update.

To this end, numerous attemps have been made by researchers and network operators [5]–[7], and RPKI is the most popular one among them, which has gained relatively wide deployment according to recent reports [10]. RPKI is a specialized public key infrastructure designed to improve the security of BGP, which consists of two main components, ROA and ROV [8]. The owners of a prefix p can register in ROA to get the records that state which ASes are authorized to originate p, which has cryptographic certificate as technical basis and RIRs (which are in charge of allocating prefixes) as trust anchors. Each ROA record contains an ASN A and a prefix p which means A is authorized to advertise p, and a max-length field denoting that prefixes longer (more specific) than max-length is considered invalid even if it's a sub-net of A. It's initially designed to reduce the space required to store the ROA records and make it more flexible to reconfigure networks without modifying the ROA records, however recent studies show that it often incurs misconfigurations and has potential vulnerabilities [20]. ROA records are stored in a centralized database named RPKI repository.

Then upon receiving an update, the ASes which adopt ROV fetch ROA records from RPKI repository, and validate the update according to the records. If there are records authorizing the AS in the update to advertise the prefix, it is considered as valid; if there are only records authorizing other ASes to advertise the prefix, it's invalid, and if there are no records for the prefix, the update is considered unknown (as RPKI has not been fully-deployed so far). The valid and unknown updates get propagated as normal, but the invalid updates are filtered. Fig. 1(b) shows how RPKI avoids the propagation of invalid updates. Suppose prefix p is registered in ROA and AS 2 is deployed with ROV, when AS 2 receives both the valid and invalid updates, it can fetch ROA records from RPKI repository, which shows that only AS 1 is quantified to originate p, and filters the update from AS 666 accordingly.

C. Existing Measurement of RPKI

Given the importance of RPKI in securing BGP from hijacking, many studies have been devoted to measuring the deployment of RPKI, which mainly aim to address two concerns: (1) **How many prefixes have been registered in/proctect by ROA**, and (2) **How many ASes are adopting ROV to filter invalid updates**. As for concern (1), researchers can analyze all the ROA records, and compare it to the global RIB (Route Information Base) table from public route collectors, and the results can be derived directly.

Measurement of ROV. Unlike concern (1), things get much harder for concern (2). There are no available data to show which AS is adopting ROV, so experiments have to be conducted to see which ASes are actually filtering invalid updates. Early measurement [9] proposes to observe the propagation of invalid updates from global RIB table got from public route collectors. However, this method can easily get confused with other filtering and is heavily influenced by the distribution of vantage points [13], resulting in low accuracy. A more accurate method is to announce an invalid prefix to the target AS and check whether it is accepted by investigating the RIB table (if available) or testing the reachability of the prefix from a probe in the target AS. [13] uses this method to conduct the measurement, and includes a valid prefix as a control group to distinguish from other filtering. However, this method has two main drawbacks: First, the target AS has to peer with PEERING (this problem is solved by the inference technique introduced below); Second, PEERING can only advertise prefixes from several certain sites. These two restrictions greatly narrow down the measurement range, which makes it only capable of identifing the ROV adoption of $\sim$100 ASes.

Inference techniques. As mentioned above, one major limit of [13] is that it requires the target AS to peer with PEERING testbed, and the inference technique proposed by [14] have solved this problem. Instead of observing whether a certain AS filters the invalid update, it collects updates from public route collectors to see which **paths** are filtering invalid updates and which paths are not. After labeling the paths, it models the problem in a probabilistic setting: it

infers the probability distribution of the ASes deploying ROV, which changes the output from a binary value in $\{0, 1\}$ to a distribution on $[0, 1]$, and then it treats the labeled paths as observed data and uses a famous Bayesian inference technique named Markov Chain Monte Carlo (MCMC) to solve the problem. However, MCMC uses a sampling method to appoximate the probability distribution, which is extremely time-consuming and becomes unscalable when the problem becomes larger, and sometimes may be hard to access convergence [17].

III. ROV Measurement with ROV-MI

This section first introduces the challenges for performing a large-scale, accurate and efficient ROV measurement, and our solutions for them (§ III-A), and then presents the framework of ROV-MI (§ III-B).

A. Challenges and Solutions

As analyzed in § II-C, the major challenges in measuring ROV deployment can be summarized as follows:

Challenge 1: Limited measurement range. As previous studies can only announce prefixes from several sites of PEERING testbed, even with sophisticated inference techniques, the measurement range is still limited by the number of labeled paths.

Challenge 2: Inefficient or inaccurate inference techniques. Simple heuristic used in early works have low accuracy. Recent attempts solve it more accurately in a probabilistic setting through a sampling-based Bayesian inference method, but the method is extremely time-consuming and unscalable.

To step over the barrier in current measurement of ROV, we propose the solutions to the challenges above:

Solution 1: Utilizing in-the-wild invalid prefixes. Through previous analysis, we identify that the key reason of limited measurement range is that PEERING testbed provides **only a few origins to advertise invalid prefixes**, which results in that only a small number of paths are labeled about whether they propagate invalid updates or not. Our solution is to **utilize in-the-wild invalid prefixes to increase the number of the origins of invalid updates**. Fig. 2 shows how many unique RPKI invalid prefix-origin pairs are observed from global updates collected from public route collectors during May, 2021. It can be seen that over 6,000 unique invalid prefix-origin pairs can be observed every day, which also conforms to observation of NIST RPKI monitoring platform [21]. This is much larger compared to the number of sites provided by PEERING testbed to advertise invalid updates, which provides the possibility of a large-scale ROV measurement. After getting enough origins of invalid updates, the next question is how to observe the propagation of these prefixes, so as to label the paths. To this end, we propose a data-plane active probing scheme to identify which paths are filtering invalid updates: we use probes to traceroute two IP addresses from a legitimate (valid or unknown)/invalid prefix pair of the same origin, to get the AS-level paths, and through the comparison of these paths, we know whether these paths are filtering invalid updates.

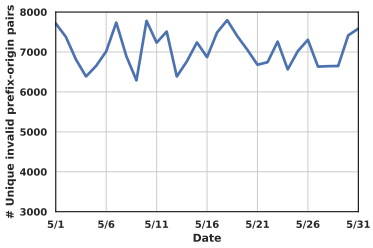


Fig. 2: The number of unique RPKI invalid prefix-origin pair observed in global updates during May, 2021. Over 6,000 unique invalid prefix-origin pairs are observed per day, which provides the possibility to use in-the-wild invalid prefixes for ROV deployment measurement.

Solution 2: Accurate and Efficient Bayesian inference techniques. As have been shown by experiments, using heuristics for inference of ROV deployment can be severely affected by the distribution of the vantage points and easily confused with other types of filtering, so we decide to borrow the idea proposed by [14] to model the problem in a probabilistic setting for a more accurate inference. However, existing methods like MCMC have to randomly generate samples to approximate the distribution, which is quite inefficient, and unscalable when the problem becomes large (as it is in a large-scale measurement). So the key problem is how to solve it both accurately and efficiently. To achieve this, we utilize an advanced Bayesian Inference method named Stein Variational Gradient Descend (SVGD) proposed in [18]. SVGD starts with a group of initial particles, and iteratively update the particles to minimize the difference between the approximate distribution (derived from the particles) and the target distribution. The biggest advantage of SVGD is that it provides a deterministic descent method, unlike MCMC, which uses a random method to draw samples from the target distribution thus requiring large sample size to increase the diversity of particles. This advantage makes SVGD significantly superior to MCMC in terms of efficiency.

B. Framework

After giving corresponding solutions to the challenges of ROV measurement, we introduce the design of ROV-MI. Fig. 3 shows the framework of ROV-MI. In the measurement part, we use the data from public route collectors to detect the occurrence of invalid prefixes in global updates in real time. After spotting an invalid prefix announcement from an AS, we find another legitimate prefix that has the same origin AS with it, and respectively traceroute a live IP address in these two prefixes with probes. By comparing the AS-level path obtained from the traceroute, we can label whether each path is filtering invalid updates or not.

Then in the inference part, we consider the problem under a probabilistic model: we can regard the labeled paths

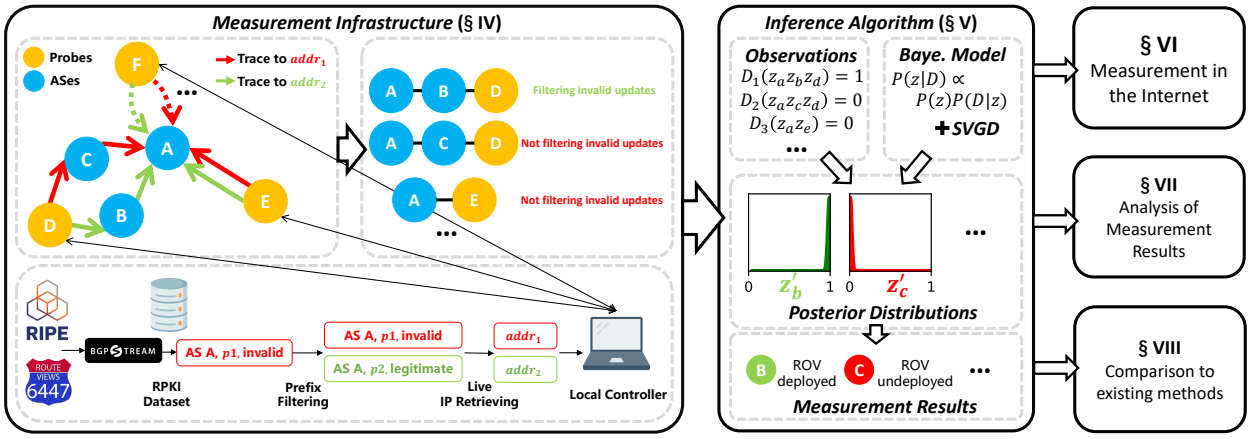


Fig. 3: The framework of ROV-MI.

as observed data, and our task is to infer the posterior distribution of the ROV deployment of each AS given these data. Next, we use SVGD to solve this problem efficiently and accurately.

IV. Measurement Infrastructure

The workflow of ROV-MI can be intuitively described as **finding invalid prefixes on the control plane, and labeling the paths on the data plane**. First of all, in the control plane, we collect the BGP updates in the global Internet in real time, extract the prefix and origin AS and check whether they are valid against ROA records. If we spot an invalid pair of prefix p_1 and origin AS A, we find another legitimate prefix p_2 from the updates which also originates from AS A, and the validity of this prefix is valid or unknown (which means it will not be filtered by ROV). Then in the data plane, we first respectively retrieve an active IP address from p_1 and p_2, denoted as $addr_1$ and $addr_2$, and use probes to perform traceroutes towards this pair of IP addresses, to find the path chosen by the probes to reach $p1$ and $p2$. Finally, by comparing the AS-level path obtained by the traceroutes, we can label which paths are filtering invalid updates. The rest of this section mainly introduces several main modules of the measurement infrastructure in detail and discusses about the ethics of the measurement.

A. Spotting Invalid Prefixes

In order to find invalid prefixes in the control plane, the first step is to collect all the BGP updates, and we achieve this with BGPStream [22]. BGPStream is an open-source software that provides an interface for real-time access to global updates. The data sources of BGPStream include RIPE RIS (Route Information Service) [23] and RouteViews [24], which are two projects dedicated to providing network operators and researchers with BGP information in the global routing system. The working principle of these two projects is to maintain some route collectors to establish BGP connections with the routers from volunteering ASes, so as to obtain the updates advertised by these routers and make them public. Until June 2021, RouteViews and RIPE RIS have maintained 37 and 26 route collectors respectively. These route collectors have established connections with more than one thousand ASes, providing high visibility for global updates.

After obtaining the available updates, we need to identify the invalid ones from them. There are many open source RPKI validators that can accomplish this work. RPKI validators allow users to periodically download ROA records from the RPKI repository, and then users can compare the downloaded data with received updates, or feed them to the router for processing. In our measurement infrastructure, we select Routinator [25] as the RPKI validator and set the time interval for synchronizing data to 10 minutes, as suggested in previous work [11]. Then for each update we receive with BGPStream, we validate it against the ROA records, and the invalid ones are used for measurements.

B. Prefix Selection and Live IP Retrieving

Directly using all invalid prefixes for further measurement will be interfered by some other factors (e.g. multi-homing prefixes), so we need to filter some invalid prefixes to exclude these factors to make our measurement more accurate. After that, for each invalid prefix, we need to find a legitimate prefix of the same origin, and extract a live IP addresses from this prefix pair.

Filtering of multi-homing prefixes. Multi-homing prefix refers to the prefix announced by multiple ASes in the control plane, which may be caused by many reasons, such as BGP Anycast, service migration, route hijacking, and so on [26]. In this case, the accessibility and path to the prefix may be unstable, and the results obtained by different probes may be inconsistent, which is not convenient for our measurement.

We filter multi-homing prefixes by maintaining a prefix-trie (we also use the same data structure for other prefix filtering and live IP extraction). When an update is received, we will insert the prefix that it contains into the prefix-trie, and add the ASN to the attributes (maintained through a list) of the end node of the prefix in the trie. Then after finding an invalid update in this way, we can query the invalid prefix in it on the trie, if there are attributes other than the ASN in the update, the prefix is a multi-homing prefix.

Filtering of prefixes covered by other legitimate prefixes. Some invalid prefixes are caused by exceeding the

max-length and in this case, it may be covered by another shorter (less specific) but legitimate prefix. In this case, the invalid prefix may still be reachable on a path which filters invalid updates, because this shorter but legitimate prefix can still propagate along this path.

The filtering of covered prefixes is similar to the filtering of multi-homing prefixes. First, we use the same method to maintain a prefix-trie of legitimate prefixes. Then if we want to query an invalid prefix, we can move from the root node, along its ancestor nodes to its own node on this trie. In this process, if one of its ancestor nodes have an attribute, it means that this invalid prefix is covered by another legitimate prefix.

Live IP Retrieving. The invalid prefix which passes the filtering will be used for subsequent measurement, so the next task is to find a legitimate prefix of the same origin for the invalid prefix, and extract live IP addresses from this pair of prefixes. Finding the legitimate prefix is relatively simple. We only need to maintain a mapping from ASN to legitimate prefix when receiving the updates, and query it when necessary. Finding the active address is a bit more complex. It is too time-consuming to find the address by random scanning, so we periodically scan all IPv4 addresses with ZMap [27] (because it is difficult to extract live IP address for IPv6 prefixes, we only consider IPv4 prefixes in our work), and maintain a live-IP-trie to store the detected live IP addresses. Then to retrieve a live IP address of a prefix, we only need to check whether there is a subtree rooted at the corresponding node of the prefix. If there is, each leaf node in the subtree represents a live IP address of the prefix.

Influence of other filtering. To rule out the influence of other filtering, we confirm that the updates used for further experiments we receive do not trigger other types of filtering (e.g., Route Flap Damping, Loop Detection [14]). There are also some common filtering which are similar to RPKI in mechanism (DISP (Drop If Still Routable) [28], BGPSec [7], etc.), but they either are designed for invalid prefixes that exceeds the max-length, which would not interfere with our measurement, or have limited deployment, which has minor influence on our measurement results.

C. Traceroute and Labeling the Paths

In the data plane, we perform traceroutes with the probes towards the live IP extracted from the legitimate/invalid prefix pair, and compare the paths obtained by the traceroutes to infer whether they filter invalid updates.

Traceroute with public probes. There are many public probe services, and we choose RIPE Atlas [29] and perfSONAR [30] from them to perform our measurements. RIPE Atlas is a global active measurement network from RIPE NCC. Participants of RIPE Atlas can deploy their own probes with dedicated hardware devices or software agents provided by RIPE NCC, and all the probes are public. Users can use any probe to make their customized measurement of different types (i.e. ping, traceroute, DNS, etc.). Similarly, perfSONAR is another network measurement toolkit aiming

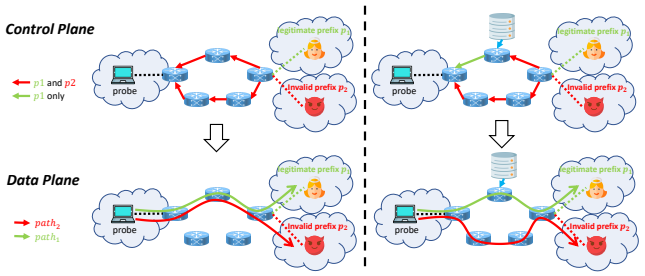


Fig. 4: An example of path labeling. The left part shows the case where the probe has identical paths to the legitimate and invalid prefixes, and the right shows the case of the different paths.

to provide federated coverage of network paths, which can also be deployed by volunteers as probes and publicly accessed by the users. perfSONAR supports a wider range of measurement types compared to RIPE Atlas (i.e. path, packet loss, throughput, etc.). There are ∼25,000 probes in RIPE Atlas and ∼1,900 pScheduler servers in perfSONAR respectively by June, 2021.

Labeling the paths. After launching the traceroutes on the data plane, we can label which paths filter invalid updates by comparing the paths from the probes to the invalid/legitimate prefixes. Assuming that we traceroute the invalid prefix and the legitimate prefix from one probe, and the AS-level paths obtained are denoted as $path_1$ and $path_2$ respectively, then there are two different situations: The first is that $path_1$ and $path_2$ are the same. As shown on the left of Fig. 4, in the control plane, this path must have propagated the invalid update, otherwise the probe cannot choose $path_1$ to reach the invalid prefix. So in this case, we can think that this path does not filter invalid updates. The second case is that $path_1$ and $path_2$ are different. As shown on the right of Fig.4, firstly, $path_2$ cannot have filtered the invalid update, otherwise it is impossible for the probe to reach the invalid prefix through $path_2$. Secondly, $path_1$ filters the invalid update, because if the probe receives the update from $path_1$, which means the prefix is reachable from both $path_1$ and $path_2$, then the probe will choose the better route $path_1$ for routing just like it does for the legitimate prefix, which is inconsistent with the traceroute result. So in this case, we think that $path_1$ filters invalid updates, and $path_2$ does not filter invalid updates (here we assume no traffic engineering for the invalid/legitimate prefix pair).

D. Ethics

In our measurement infrastructure, we do not announce any messages in the control plane, so the possible ethical issue lies in the scanning of the live IP addresses and the traceroute to the live IP addresses.

First for the live IP scanning, we limit the bandwidth of the scan to 100Mbps and only execute the scanning on daily basis, so it doesn't cause much burden on the Internet. For the detected live IP addresses, we do not take any further

actions except performing a traceroute towards it when it's contained in an invalid/legitimate prefix pair. Also, we do not analyze the live IP addresses or make them public, so there are no risks for the leakage of their privacy.

Then in the traceroute, we control the rate of performing traceroute to 1 time per second, so it does not cause any burdens on the Internet either.

V. Inference Algorithm

This section introduces how ROV-MI infers the adoption of ROV with the labeled paths from the measurement infrastructure. We first explain how to convert the problem into a Bayesian inference problem, and then introduce how to solve it through an advanced Bayesian inference method, SVGD.

A. Non-probabilistic Model

First, we formalize the problem of ROV deployment inference under a non-probabilistic model. For an AS i, we use a variable x_i to indicate whether it deploys ROV. For the simplicity of calculation, we set the meaning of x_i to AS i **not adopting** ROV. x_i is equal to 0 when AS i adopts ROV and 1 otherwise, which can be expressed mathematically as:

$$x_i = \begin{cases} 0 & \text{if AS } i \text{ adopts ROV} \\ 1 & \text{if AS } i \textbf{ does not} \text{ adopt ROV} \end{cases} \tag{1}$$

Then in our measurement, we cannot directly observe which ASes are adopting ROV, and we only measure which paths are filtering invalid updates. For a path j, we denote the ASes it contains as N_j. We notice that path j filters invalid updates when there is at least one AS in N_j filters invalid updates (i.e. adopts ROV) on it . So we can use the production of $\{x_i\}_{i \in N_j}$ to determine whether the path j filters invalid updates. Similarly, we use a variable y_j to denote whether path j filters invalid updates, where:

$$y_j = \begin{cases} 0 & \text{if path } j \text{ filters invalid updates} \\ 1 & \text{if path } j \textbf{ does not} \text{ filter invalid updates} \end{cases} \tag{2}$$

Then the relationship of the ROV deployment of the ASes and filtering of the paths can be expressed as:

$$y_j = \prod_{i \in N_j} x_i \tag{3}$$

For each observed path, there is an equation in the form above, so the results of our measurement can be regarded as a set of equations. If the equation set is solved, the deployment of ROV can be inferred. However in reality, solving this equation set is nontrival due to the following reasons:

1) It is difficult to obtain a unique solution [14]. Generally, finding a unique solution to such a equation set requires a large number of equations, which means that a lot of independent measurement results need to be obtained to make accurate inference, which is usually hard to meet. In other problems, there may be some additional assumptions to assist solving the equations (e.g., in the inference of link failure, it can be assumed that the failure is very sparse, thus limiting the number of x_i with the value of 0), but it is not feasible in ROV deployment measurement.
2) The solution is vulnerable to noises. Under most circumstances, there are some noises in the measurement results. In this case, this equation set is highly likely to have no solution. Some robust methods will approximate the solution by introducing some Gaussian noises into the model, but this is not suitable for ROV deployment inference, because the noise in the measurement usually comes from a small number of incorrect labels, which cannot be modeled with Gaussian noises.

B. Probabilistic Model

In lights of the problems of non-probabilistic model, we choose to model the problem in a probabilistic setting [14]. In order to introduce probability model, we use a new variable z_i to represent **the percentage of invalid routes that AS i does not filter**. Then different from the x_i in the non-probabilistic model, which is a binary variable with its value in $\{0, 1\}$, z_i is a random variable distributed in $[0, 1]$. When the value of z_i is in $(0, 1)$, we can interpret it as AS i will filter the received invalid updates with the probability of $(1 - z_i)$. Assuming z_i is inter-independent in its *prior* distribution, we can derive the relationship between z_i and y_j with the following method: when y_j is equal to 0, it means that path j does not filter the invalid update, that is to say, **all** the ASes on path j don't filter the invalid update, otherwise y_j is equal to 1. Therefore, the probability that y_j is equal 0 can be expressed as:

$$\begin{cases} p(y_j = 0 \mid \boldsymbol{Z}) = \prod\limits_{i \in N_j} z_i \\ p(y_j = 1 \mid \boldsymbol{Z}) = 1 - \prod\limits_{i \in N_j} z_i \end{cases} \tag{4}$$

where:

$$\boldsymbol{Z} = (z_1,\ z_2,\ \ldots,\ z_n) \tag{5}$$

Here n represents the number of ASes to be inferred. Then for a set of measured data $D = (y_1,\ y_2,\ \ldots,\ y_N)$ (where N denotes the number of labeled paths), the probability of observing D given a value of $\boldsymbol{Z}$ is:

$$p(D \mid \boldsymbol{Z}) = \prod_{y_j in D} p(y_j \mid \boldsymbol{Z}) \tag{6}$$

The equation Eq. (6) can be regarded as a *likelihood* function (a function of $\boldsymbol{Z}$), which represents the probability that the data D is observed when $\boldsymbol{Z}$ takes different values. The problem we now hope to solve is to calculate the probability distribution of $\boldsymbol{Z}$ under a given set of measurement data, namely the *posterior* distribution of $\boldsymbol{Z}$. Bayesian rule tells us the *posterior* probability is proportional to the product of the *prior* probability and the likelyhood function:

$$p(\boldsymbol{Z} \mid D) \propto p(\boldsymbol{Z}) p(D \mid \boldsymbol{Z}) \tag{7}$$

By combining Eq. (4)-Eq. (7) and providing a *prior* distribution $p(\boldsymbol{Z})$ (which can incorporate known information about ROV deployment, and if there is no *prior* knowledge about ROV deployment, some common distributions like uniform distribution can be used), the *posterior* distribution can be determined. Next, in order to determine whether AS i deploys ROV, we need to solve the marginal distribution $p(z_i|D)$ of the corresponding dimension z_i in $\boldsymbol{Z}$. However, it is intractable to give an analytical solution of $p(z_i|D)$ because $\boldsymbol{Z}$ is a random variable located in a high-dimensional space (the dimension n is equal to the number of inferred ASes), and different dimensions are highly correlated. So we need to use Bayesian inference methods to approximate the marginal distributions. However, poor man's Bayesian estimator methods like MAP (Maximum a *posterior*) provide limited information about the distribution, while full Bayesian inference technique like Variational Inference or MCMC have accuracy or efficiency issues for high dimensional data, so it is essential to find an inference method to solve the problem both accurately and efficiently.

C. ROV Deploymenet Inference with SVGD

The MCMC method used in previous work uses sampling to solve $p(\boldsymbol{Z}|D)$: it starts from an initial state $\boldsymbol{Z}^0$ and then iteratively draw samples from $p(\boldsymbol{Z}|D)$. In the t-th iteration, it generates a new distribution $q^t(\boldsymbol{Z})$ based on the samples generated in the previous $(t-1)$ iterations (for example, a new normal distribution with the mean of $\boldsymbol{Z}^{(t-1)}$) and randomly draws a new sample $\boldsymbol{Z}_{temp}$ from it, and accepts it with the probability of $\alpha(\boldsymbol{Z}^0, \ldots, \boldsymbol{Z}_{temp})$. If $\boldsymbol{Z}_{temp}$ is accepted, it makes $\boldsymbol{Z}^t \leftarrow \boldsymbol{Z}_{temp}$ and starts the $(t+1)$-th iteration, otherwise it repeats the t-th iteration. $\alpha(\cdot)$ is a self-defined function that controls the direction of sampling, which usually makes the sampling method to draw more samples in the region with higher probability density. After the sampling is completed, the set of all the accepted samples $\{\boldsymbol{Z}^0, \ldots, \boldsymbol{Z}^T\}$ can be regarded as the approximate distribution of $p(\boldsymbol{Z}|D)$.

The main problem with MCMC lies in that it is based on ***a random scheme***: In each iteration, it randomly draws an sample, and then accepts it with a certain probability, which means that it may need to sample $\boldsymbol{Z}_{temp}$ for many times to obtain a new accepted sample $\boldsymbol{Z}^t$. And in order to better approximate the distribution, the diversity of samples must be increased to ensure that the value space of the random variables is fully explored, so a larger number of samples is required, especially for high-dimensional random variables like $\boldsymbol{Z}$. This makes MCMC very time-consuming, unscalable for large problems, and have accuracy and convergence issues when the sample size is not sufficiently large.

To this end, we propose to use a more advanced Bayesian Inference algorithm, SVGD to solve the approximate distribution of $p(\boldsymbol{Z}|D)$. SVGD is also a particle-based method, which means that it also samples from $p(\boldsymbol{Z}|D)$ to approximate the distribution. But unlike MCMC, SVGD is based on a ***deterministic*** method: First, through Stein method [18], SVGD **gives an analytical solution of the direction of the gradient** for updating the particles (samples). Based on this gradient, we can use the deterministic gradient descent method to iteratively update the particles, which can effectively increase the speed and efficiency of optimization and does not require large sample size to increase the diversity of particles like random methods.

As shown in Algorithm 1, the process of solving $p(\boldsymbol{Z}|D)$ with SVGD is as follows:

First, we randomly generate a set of initial particles $\{\theta_i^0\}_{i=1}^m$ (the number of particles is fixed during the iteration) for subsequent optimization. In the l-th iteration, we first calculate the direction of the gradient for updating the particles:

$$\hat{\phi}^*(\theta) = \frac{1}{m}\sum_{k=1}^{m}[k(\theta_k^l, \theta)\nabla_{\theta_k^l}\log p(\theta_k^l|D) + \nabla_{\theta_k^l}k(\theta_k^l, \theta)] \tag{8}$$

Kernel Functions for SVGD. The $k(\cdot,\cdot)$ term in Eq. (8) is called kernel function, and the choice of the kernel function may influence the convergence speed of the algorithm, and is generally tested via experiments. In our work, we mainly consider two types of kernel functions:

- Gaussian(RBF) kernel:

$$k(x, x') = exp(-\frac{1}{h}||x - x'||_2^2) \tag{9}$$

- Inverse Multi-quadric(IMQ) kernel:

$$k(x, x') = (1 + \frac{1}{h}||x - x'||_2^2)^{-\beta} \tag{10}$$

In the kernel functions above, h and β can be adjusted to optimize the efficiency of SVGD, and we provide model calibration experiments in § VIII-A.

Intuitively, the first term in Eq. (8) makes the particles move towards the region of higher probability density, and the second term acts like a regularization term that forces the particles to move away from current value to avoid local optima. Note that in Eq. (8), with a certain kernel function, we only need to calculate $\nabla_{\theta_k^l}\log p(\theta_k^l|D)$ to determine the gradient function, which can be calculated as:

$$\begin{aligned}\nabla_{\theta_k^l}\log p(\theta_k^l|D) &= \nabla_{\theta_k^l}\log p(\theta_k^l)p(D|\theta_k^l) \\ &= \nabla_{\theta_k^l}\log[p(\theta_k^l)\prod_{y_j\in D}p(y_j|\theta_k^l)] \\ &= \nabla_{\theta_k^l}\log p(\theta_k^l) + \sum_{y_j\in D}\nabla_{\theta_k^l}\log p(y_j|\theta_k^l)\end{aligned} \tag{11}$$

In Eq. (11), the first term is only related to the prior probability, so in order to solve $\nabla_{\theta_k^l}\log p(\theta_k^l|D)$, we only need to traverse all y_j, calculate $\nabla_{\theta_k^l}\log p(y_j|\theta_k^l)$ and sum them up. And we can see that solving $\nabla_{\theta_k^l}\log p(y_j|\theta_k^l)$ is easy from the form of $p(y_j|\theta_k^l)$ in Eq. (4).

After getting the gradient function, we can use it to update the particles:

$$\theta_k^{l+1} = \theta_k^l + \epsilon_l\hat{\phi}^*(\theta_k^l),\ j = 0, 1, \ldots, m \tag{12}$$

where ϵ_l is the step size of the l-th iteration. After T iterations, we obtain the particles required to approximate the distribution.

Algorithm 1 Bayesian Inference with SVGD

Input The prior distribution $p(\boldsymbol{Z})$, the initial particles $\{\theta_k^0\}_{k=1}^m$, and the max iteration time T
Output The final set of particles $\{\theta_k^T\}_{k=1}^m$ that approximate the posterior distribution $p(\boldsymbol{Z}|D)$

for $l \leftarrow 0$ **to** T **do**
  $\triangledown$ *compute the gradient function*
  $\hat{\phi}^*(\theta) = \frac{1}{m}\sum_{k=1}^{m}[k(\theta_k^l,\theta)\nabla_{\theta_k^l}\log p(\theta_k^l|D) + \nabla_{\theta_k^l}k(\theta_k^l,\theta)]$
  for $k \leftarrow 1$ **to** m **do**
    $\triangledown$ *update the particles*
    $\theta_k^{l+1} = \theta_k^l + \epsilon_l\hat{\phi}^*(\theta_k^l)$
return $\{\theta_k^T\}_{k=1}^m$

Accelerating the algorithm. From Eq. (11), we can find that the main computational complexity of the algorithm lies in that for each particle, we need to traverse y_j to compute $\sum_{y_j \in D}\nabla_{\theta_k^l}\log p(y_j|\theta_k^l)$. In light of this, we can use two techniques to accelerate the computation.

First, from the nature of BGP, we know that each path only contains a few ASes, which means each $p(y_j|\theta_k^l)$ is related to only a few dimensions in θ_k^l, and the components of $\nabla_{\theta_k^l}\log p(y_j|\theta_k^l)$ in other dimensions are all zero. So we can use this ***sparse relationship*** between D and θ_k^l to accelerate the algorithm. For each dimension of θ_k^l, we maintain which paths in D contribute to the component of $\nabla_{\theta_k^l}\log p(y_j|\theta_k^l)$ in this dimension, then we can compute $\nabla_{\theta_k^l}\log p(y_j|\theta_k^l)$ dimension by dimension, and for each dimension, we only need to visit the paths that is related to it.

Second, if the complexity is still very high after the optimization above, then we can **downsample** D to approximate the calculation of $\nabla_{\theta_k^l}\log p(D|\theta_k^l)$. We select a subset $S \subset \{0, 1, \ldots, N-1\}$, and then calculate it with:

$$\nabla_{\theta_k^l}\log p(D|\theta_k^l) \approx \sum_{j\in S}\nabla_{\theta_k^l}\log p(y_j|\theta_k^l) \tag{13}$$

If possible, some parallel computation methods can also be used to accelerate the process, such as paralleling the gradient calculations of different particles.

D. From Posterior Distribution to Results

The output of SVGD are the particles $\{\theta_k^T\}_{k=1}^m$ which approximate the distribution $p(\boldsymbol{Z}|D)$. Then from the definition of $\boldsymbol{Z}$, we know that the i-th dimension z_i is related to whether AS i adopts ROV. So we can use the i-th component of all the particles to approximate the marginal distribution of z_i. Note that z_i represents the probability of AS i **not** deploying ROV, so we define another variable $z_i' = 1 - z_i$, and then it is more intuitive to analyze the distribution of z_i' .

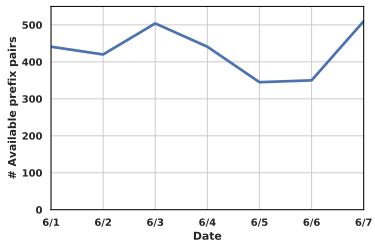


Fig. 5: The number of prefix pairs that can be used for data plane probing. Though most invalid prefixes are filtered, the number of available prefix pairs is still large enough to perform a large-scale ROV deployment measurement.

VI. Measurement in the Internet

In this section, we present our measurement results on the Internet, including the path labeling results from our measurement infrastructure and the ROV deployment inference results obtained from our inference algorithm.

A. Experiment Setup

We use ROV-MI to measure ROV deployment on the Internet from June 1st to 7th in 2021. In our measurement infrastructure, we configure BGPStream to live mode to capture the updates from 5 route collectors (route-views2-4.routeviews.org, rrc01-02.ripe.net), and upon finding an available invalid prefix, we initiate traceroutes from 200 randomly seleted public probes. In the inference part, we set the particle number in SVGD to 1,000. All the experiments were carried out on a Dell PowerEdge R720 server using Ubuntu 16.04 LTS with 6 cores Intel Xeon(R) CPU E5-2630 @ 2.60GHz and 94G RAM.

B. Results from Measurement Infrastructure

Control Plane Results. Fig. 5 records the daily number of the prefix pairs that can be used for data plane probing. We define the prefix pair as **available** if it is not filtered, and we can extract live IP addresses from these two prefixes. We can see that the number of available prefix pairs fluctuates between 300 and 500, which is an order of magnitude smaller than the daily number of invalid prefixes recorded in Fig. 2. Through the analysis of the filtering results, we found that most ($\sim$71%) invalid prefixes are filtered because they are covered by another valid prefix. This result implies that there are a large number of configuration errors about the max-length field in the ROA records, which is consistent with the analysis in related work [12]. However, this number is still large enough for us to use these prefix pairs for a large-scale ROV deployment measurement.

Data Plane Results. Next, we initiate traceroutes to these available prefix pairs in the data plane. The entire dataset get from data plane is recorded in Table II. It can be seen that compared with the measurement method in previous work, our method can obtain about ten times the paths and cover about ten times the ASes, which shows that our measurement infrastructure effectively increases the measurement range.

TABLE II. Data collected from data plane probing.

	#unique origins	#covered ASes	#unique paths
Gray *et al.* [14]	1	1,265	12,634
ROV-MI	678	11,074	115,427

What's more, most ASes that we are able to measure are large ASes, which have greater impact on the filtering of the invalid updates in the Internet. Detailed analysis of the cone-size [31] distribution of the measured ASes is shown in Appendix § C.

C. Inference Results of ROV Deployment

Typical cases of posterior distributions. First we present several typical cases of the posterior distribution to show how the distribution reflects the deployment of ROV. Fig. 6 shows the posterior distributions of z' for four typical ASes.

- **AS 6677.** The mode of the distribution is very close to 0, and the possible values of z' are clustered in a small interval. This case shows that we are confident that AS 6677 doesn't adopt ROV.
- **AS 6453.** The mode of the distribution is very close to 1, and the possible values of z' are clustered in a small interval. This case shows that we are confident that AS 6453 adopts ROV.
- **AS 47441.** The mode of the distribution is about 0.2, and the possible values of z' are also clustered in a small interval. This means that AS 47441 filters $\sim$20% of the invaild updates it receives, which indicates that it is adopting different filter policies either to different neighbor ASes or to different prefixes. So we consider it as partially-deployed.
- **AS 7296.** The possible values are spreading in a large interval, and the distribution is very similar to a uniform distribution. This result means that we have low evidence about whether this AS adopts ROV, so we consider it as unknown.

Summarizing the posterior distribution. It's infeasible to observe the posterior distributions of all the ASes and judge whether they adopt ROV one by one. So we need to use some summarizing metrics to judge ROV deployment from the distribution. We are mainly concerned with two issues: (1) The values z' is most likely to take (the probability of the AS adopting ROV). (2) For these values, what's the probability for z' to take value from them (how confident we are about the inference results). To this end, we use two summarizing metrics for quantitative judgement of ROV adoption.

- The **mean** of the distribution. By calculating the mean of the distribution, we know the expected value of z'_i which can be regarded as the probability of AS i adopting ROV.
- The **HDI** (Highest Density Interval) of the distribution. HDI refers to the smallest interval that contains γ of the mass (by default $\gamma = 0.94$). This metric estimates the spread of the distribution and shows how likely it is for z'_i to take values near the mean of the distribution, which indicates the confidence of the mean value. Since HDI and the confidence of the inference are negatively correlated, we define a new variable:

$$\text{confidence} = 1 - \textbf{HDI}$$

TABLE III. Rules for categorizing the ASes.

Categories	mean	confidence(1-HDI)
deployed	$[0.8, 1]$	$[0.5, 1]$
undeployed	$[0, 0.2]$	$[0.5, 1]$
partially-deployed	$(0.2, 0.8)$	$[0.5, 1]$
unknown	-	$[0, 0.5)$

TABLE IV. The number/proportion of different types of ASes.

Categories	Number	Proportion
deployed	3,107	28%
undeployed	4,716	43%
partially-deployed	357	3%
unknown	2,894	26%
total	11,074	100%

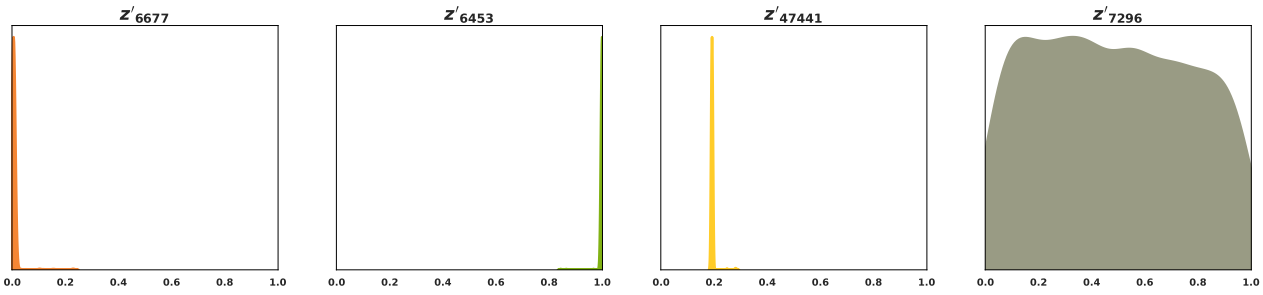
Based on these two metrics, we can set some classification rules to judge whether each AS deploys ROV. We use a simple threshold method in our inference, which can divide the ASes into four categories according to the rules in Table III. Fig. 7 shows scatter plot of a subset of ASes located in the mean-confidence space. It can be seen that most ASes of type deployed or undeployed show very high confidence about the results, which confirms the feasibility of the categorization scheme.

Reason for *unknown* ASes. Through the analysis of the variation of the gradient value with the iteration number in the runtime of SVGD as well as the labeled paths that contains the unknown ASes, we identify that the algorithm fails to infer the ROV deployment of the *unknown* ASes mainly in the following two situations:

- **Isolated path**: All the ASes on a labeled path have little overlap with other labeled paths, which results in low evidence for the ROV deployment of all the ASes on this **isolated path**.
- **Masked AS**: All the labeled paths of an AS contains another AS that adopts ROV, so it is hard to infer the ROV deployment of the **masked AS** from the filtering behavior of these paths.

Detailed analysis of the reason for the *unknown* ASes are provided in Appendix § A.

Measurement Results. The categorization for all the ASes with the rules above is shown in Table IV. We are very pleased to see that, together with ROA, the deployment of ROV is also gradually increasing. Among the measured ASes, $\sim$28% have already deployed ROV and are filtering invalid updates correctly, and most ASes deployed with ROV are large transit ASes, which means ROV is contributing more than this percentage shows, making RPKI highly potential in the future.



(a) AS 6677 (undeployed) (b) AS 6453 (deployed) (c) AS 47441 (partially deployed) (d) AS 7296 (unknown)

Fig. 6: Posterior distribution of z' for four typical ASes, whose inference results are undeployed, deployd, partial deployed and unknown respectively from left to right. The posterior distributions of z' have obivious characteristic that indicate the ROV adoption of the AS.

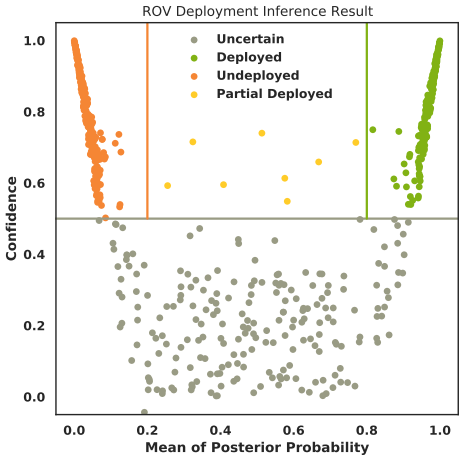


Fig. 7: The scatter plot of the ASes in the mean-confidence space. It can be seen that most of the deployed and undeployed ASes show great confidence in their results.

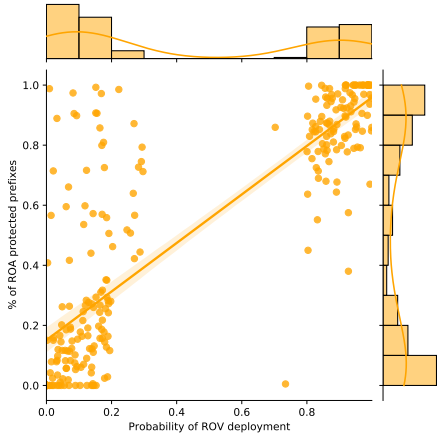


Fig. 8: The distribution of ASes in the ROA/ROV deployment plane. The deployment of ROA and ROV are highly correlated.

TABLE V. The number of ASes that deploys ROA/ROV.

ROV \ ROA	deployed	undeployed
deployed	2,261	486
undeployed	833	3,883

VII. Analysis of Measurement Results

This section mainly performs in-depth analysis of the measurement results from several aspects including the correlation of ROA/ROV deployment, the geolocation distribution of ROV deployment and the validation on the ground truth.

A. Correlation of ROA and ROV deployment

After obtaining the deployment of ROV, we can analyze the relationship between ROA and ROV deployment. For the ASes that we are able to infer the deployment of ROV, we look up the corresponding deployment of ROA from the snapshot of RPKI repository. Then we perform analysis of two different *granularities*.

First, For **coarse-grained** analysis, all the measured ASes are divided into four categories based on whether it adopts ROA and ROV (i.e., adopts both ROA and ROV/only ROA/only ROV/neither ROA nor ROV). We consider an AS deploys ROA if **at least one** of its prefixes is protected by ROA. The number of the four types of ASes are shown in Table V. Then we perform χ^2 test and get that $\chi^2 = 2988.9$, which is an extremely high value and shows that the deployment of ROA and ROV are highly relevant.

For **fine-grained** analysis, we further consider two properties of each AS: the proportion of the address space of the AS protected by ROA (which can be computed through the RIB table from global route collectors and the RPKI repository), and the **mean** of the posterior distribution z' of the AS in the inference result of ROV-MI (intuitively it is proportional to the number of the invalid updates this AS filters). These two properties respectively represent the extent to which the

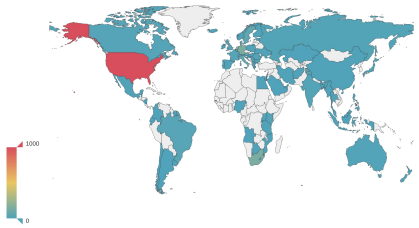


Fig. 9: The geolocation distribution the ASes that adopt ROV.

TABLE VI. The countries with the most ASes that adopt ROV.

Country	#ASes with ROV
the United States	1,171
South Africa	219
Germany	193
Netherlands	186
Italy	157

AS adopts ROA and ROV. From this view, each AS can be regarded as a sample located in a two-dimensional plane, and the distribution of the ASes is shown in Fig. 8. Then we use linear regression to analyze the correlation of these two dimensions, and get the correlation coefficient $\rho = 0.831$, which also confirms that the deployment of ROA and ROV are highly correlated.

B. Geolocation Distribution of ROV Deployment

Then we investigate whether the deployment of ROV is relevant to the geographical location. We map the measured ASes to geolocation at the national level. Then we provide the heatmap of the number of the ASes that deploys ROV around the world in Fig. 9. It can be seen that the ASes that adopt ROV is distributed in a very imbalanced way, and the countries where ROV are more fully deployed are mainly located in North America, Europe and South Africa. Then we present that countries with the most ASes that adopt ROV in Table VI. Among all the countries, the United States has the highest deployment of ROV, with over 1,000 of its ASes adopting ROV, and the countries in Europe also have relatively high deployment of ROV. Note that this imbalanced geolocation distribution might be partially caused by the imbalanced geolocation distribution of the observed paths, so a possible future work is to incorporating more probes for geoloation related analysis.

C. Validation on Ground Truth

Dataset and metrics. Finally, we verify our results on a small set of ground truth. The source of ground truth is a project called is-bgp-safe-yet [32]. This project collects the RPKI deployment status of 370 ASes by communicating with the network operators, which includes the ROV deployment information of many large ASes. For each AS, is-bgp-safe-yet records whether the AS filters invalid updates. We divide the ASes into three types accordingly: filtering, not filtering, and partial filtering (including the ASes that only filter peers). In the evaluation, for each metric(e.g., precision, recall), we compute the micro-average of the three classes as the result. We validated our inference results on this dataset and compared it to other inference methods.

TABLE VII. Validation on ground truth.

Method	Precision(%)	Recall(%)
Heuristic [9]	68	73
MCMC [14]	100	100
SVGD	100	100

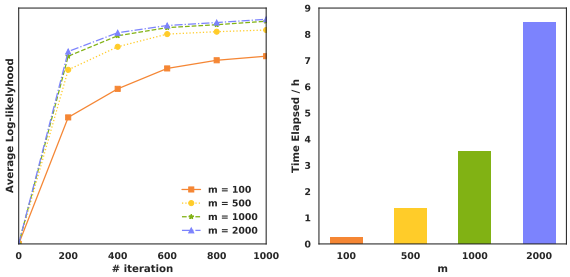


Fig. 10: The influence of the particle number m on the effectiveness and efficiency of SVGD. It can be seen that $m = 1000$ enable SVGD with both ideal accuracy and runtime performance.

Baseline Methods. We compared ROV-MI with the two baseline methods. The first is a heuristic method: It first excludes all the ASes that have propagated invalid updates. For the remaining ASes, if there is a path that filters invalid updates, it adds the confidence of all the ASes on that path by 1. Finally, the ASes with higher confidence are considered to have deployed ROV. The second baseline method is MCMC [14], a sampling based Bayesian inference algorithm which has been introduced in § V-C.

The validation results are shown in Table VII. We can see that only the heuristic method has a low accuracy, and both MCMC and SVGD can accurately infer the adoption of ROV for these ASes. But our experiments in § VIII show that SVGD is much superior to MCMC in terms of efficiency.

VIII. Comparison to other Inference Algorithms

In this section, we first evaluate the influence of the parameters on the inference algorithm and find a set of optimal parameters. Then we compare it to MCMC methods in terms of efficiency and computational costs.

A. Parameter Calibration

Influence of the particle number. The first parameter that has great impact on SVGD is the number of particles m, so we first fix the kernel function of SVGD to RBF kernel and take different values for m to observe how it influences the sampling results and runtime efficiency. The left part of Fig. 10 shows the average value of the likelyhood function of the particles (higher value indicates better sampling quality) varying with iteration times under different m. It can be

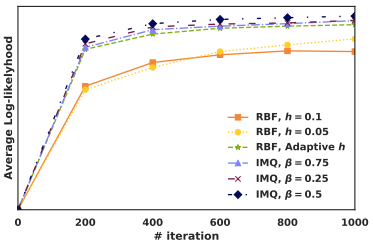


Fig. 11: The performance of SVGD under different kernel functions. It can be seen that IMQ kernel has better performance than RBF kernels, and $\beta = 0.5$ is the best setting for IMQ kernel.

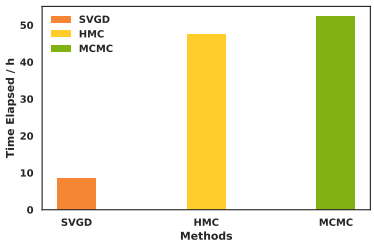


Fig. 12: The convergence time of SVGD and different MCMC methods. It can be seen that SVGD uses only ∼20% of the time as MCMC methods to reach the convergence.

seen that for m small than 1,000, enlarging m effectively improves the sampling quality, but nearly ineffective for m greater than 1,000.

The right part of Fig. 10 shows the execution time of SVGD running 2,000 iterations under different m. We can see that the execution time of SVGD increases linearly with the particle numbers m. So by combining these two experiments, we find that it is best for us to set m to 1,000 in our experiment for quality and efficiency considerations.

Comparison of different kernel functions. Another important component of SVGD is the kernel function. For each type of kernel functions mentioned in § V-C, we use several different settings to compare their performance. For RBF kernel, we use two different parameter settings of h, the first is the scheme proposed in [18], which adaptively changes the value of h according to the median value of the distance between all the particles with the equation $h = med^2/\log n$. We also try h with fixed values like $h = 0.005,\ 0.01$. Then for the IMQ kernel, we use the same scheme proposed in [18] to set the value of h, and for β, we try different values including $\beta = 0.25,\ 0.5,\ 0.75$. Similar to the experiment on m, we investigate the variation of the average log-likelyhood with the number of iteration. The results are shown in Fig. 11. First for the RBF kernel, it can be seen that fixed h have relative poor performance, and may even cause the log-likelyhood to drop with the increase of iteration number, which indicates that a fixed h is not a good choice for RBF kernels. Then for IMQ kernels, we see that they have similar performance under different parameters, and $\beta = 0.5$ is the best among the tested parameters. Finally, for the comparison of RBF and IMQ kernel, it can be seen that under all different values, IMQ kernel performs better than RBF kernel, which indicates that IMQ kernel is more suitable for our task, and we choose IMQ kernel with $\beta = 0.5$ and adaptive h as our setting for the measurement tasks and subsequent experiments.

B. Comparison to MCMC

Then we compare our inference method with MCMC. There are two commonly used MCMC methods, the original MCMC and HMC (Hamiltonian Monte Carlo). HMC replace the random walk scheme in original MCMC with a momentum based state transition, which can sample the state space more efficiently.

TABLE VIII. Number of samples when reaching convergence.

Method	SVGD	HMC	MCMC
#Particles	1,000	463,569	573,645

Convergence time. First we compare the time it takes for them to converge. In order to know the convergence time, we need some criteria to judge whether they converge. It is relatively easy for SVGD that it converges when all the particles become stable, so we can set a threshold δ, and stops the iteration when $||\theta_k^{l+1} - \theta_k^l||_2 < \delta$. But usually it is hard to tell whether MCMC converges because it is a random process. A relatively reliable method is Gelman-Rubin Method [33], which runs multiple chains that are initialized with different initial values, and judge the convergence by the statics of these chains. Through the methods above, we can determine the convergence time of the two inference methods, and the results are as shown in Fig. 12. While it only takes about 9 hours for SVGD to reach the convergence (i.e. finish the inference), solving it with MCMC methods may take ∼50 hours, which shows that SVGD is much more time-efficient than MCMC methods.

The number of samples generated. Table VIII records the number of particles generated by each method until it reaches the convergence. It can be seen that MCMC methods need to generate ∼500 times more particles than SVGD to approximate the distribution, which can cause great computational overhead.

Comparison on higher-dimensional space. As both SVGD and MCMC can reach 100% accuracy in the validation dataset, we use the synthesized data generated from the simulation on the AS topology computed from AS relationship dataset [31]. We randomly choose 5,000 ASes to deploy ROV and label sufficient paths between arbitrary origin/destination AS pairs, and then compare the precision and recall of SVGD and MCMC on the "Deployed" ASes. The results are shown in Table IX. It can be seen that both SVGD and MCMC can reach 100% in preicion, but has relatively low recall, which may be caused by **isolated paths** or **masked ASes** during the data synthesizing. However, the

TABLE IX. Experiments on higher-dimensional data.

Method	Precision(%)	Recall(%)
SVGD	100	92.98
MCMC	100	92.32

recall of SVGD is 0.66% higher than MCMC, which means that there are several ASes that MCMC fails to infer while SVGD successfully infers the deployment. This indicates that MCMC may have failed to reach convergence on the corresponding dimensions [17] and the result shows that SVGD is more *stable* to the change of the problem scale.

IX. Discussion

In this section, we discuss several design choices, limitation, and potential future works.

Inference for links. If we don't consider the measurement part, it is a good choice to do link-wise inference, because it can make more accurate inferences about behaviors such as using different filtering policies for different neighbors. However, the number of labeled paths required for precise link inference is much larger than that required for AS-wise inference, which is not feasible for our measurement infrastructure.

Incorporating IPv6 prefixes. As mentioned in § III, extracting live IP addresses from IPv6 prefixes are time-consuming and has a low hit rate [34], which is unsuitable for a real-time measurement, and by tracerouting only IPv4 prefixes can also generate enough labeled paths for inference, so we do not consider IPv6 prefixes. In future work, we can utilize some IPv6 reconnaissance techniques for IPv6 live IP address retrieving to incorporate some IPv6 prefixes and analyze how ROV deployment differs in IPv4/IPv6.

Scheduling the probes in the data plane. Generally, randomly choosing probes can cover more ASes in the labeled paths by strengthen the diversity of the probe and origin pairs. However, as discussed in § VI-C and Appendix § A, it will also result in situations like **isolated paths** and **masked ASes**. A possible improvement is to incorporate the AS topology to schedule the probes to perform traceroutes, which may provide the chance of reducing "unknown ASes" to increase the measurement coverage and generating as few labeled paths as possible for effective inference thus further boosting the performance of the inference part.

Continuous measurement. Currently with more and more ASes deploying RPKI, RPKI is developing rapidly, thus requiring measurement of longer time period. Luckily many studies and reports have shown that most invalid prefixes in the wild are *long-lived* [21], [35], which provides us the opportunity for a continuous measurement. In the future, we can make regular measurement to see how the deployment of RPKI varies with time.

Extending to other scenarios. In fact there are many problems similar to the measurement of ROV deployment like the localization of link failures and the measurement of other network properties. It is feasible to extend our inference algorithm to these scenarios. We provide the possible situations to apply ROV-MI and several examples in Appendix § B.

X. Related Work

The most related works have been introduced in § II and § III, so in this section, we briefly discuss other related works from several aspects.

RPKI Measurement. Recent years, with the gradual prevalence of RPKI, many studies have been devoting to measuring the deployment of RPKI [9], [10], [12]–[15]. The measurements for ROA deployment is mainly conducted by analyzing the ROA records in RPKI repository [9], [10], [15], which mainly aims to figure out how many prefixes/ASes are protected by the ROA as well as some ROA configuration errors and their consequences in the Internet. In terms of the measurement of ROV deployment, there are several work trying to pinpoint the adoption of ROV [9], [13], [14], but the measurement range was limited and the inference methods are unsatisfactory in either accuracy or efficiency. We extend these prior works by modifying the measurement infrastructure to enable large-scale measurement, and proposing an accurate and efficient inference algorithm based on the probabilistic model in [14].

Bayesian Inference. There is a rich literature about Bayesian inference [17]. The general purpose of Bayesian inference to estimate or approximate a target distribution. Bayesian inference techniques can be divided into two categories based on the question it aims to solve. Poor man's Bayesian estimator like MLE (Maximum Likelyhood Estimator) only gets a value of the random variable that maximize the likelyhood. We don't choose this kind of method because the information it provides is not sufficient to infer ROV deployment (there are no information like certainty). The other category, full Bayesian inference, tries to approximate the target distribution. There are two classic methods to do so, namely MCMC [36] and variationoal inference [37]. Variational inference use a distribution of certain form to approximate the target distribution, which may have accuracy issues unless the form is specially desgined. MCMC is accurate most of the time but the random scheme makes it inefficient and hard to access convergence. The SVGD [18] method combines the particle-based scheme with deterministic descend updating method, which can solve the inference problems in a accurate and efficient way.

Security Enhancement for BGP and RPKI. Except for RPKI, there are many other works proposing to provide an authentication mechanism for BGP [5]–[7], while some other works propose to protect BGP from hijackings with anomaly detection and mitigation methods [26], [38]. With the growing of RPKI, the security issues of RPKI itself have also attracted concerns from the researchers. There are several work aiming to solve the vulnerability of RPKI, including the max-length security issues [20], the configuration errors [39], and the vulnerability of the communication protocols

[40]. These works are orthogonal to our work, which all aim to make BGP reliable and robust to possible hijackings.

Binary Network Tomography. As introduced in § IX, our inference algorithm can be generalized to a class of problems called Binary Network Tomography, which aims to pinpoint the binary property of the node through path observations. Early work in binary network tomography all consider the problem in a non-probabilistic model, and solves them with mathmatical equations [41], [42]. More recent work translate the equations into logical constraints, and then solve them with SAT [43]. This type of solution have a problem in common that there might be no solution or no unique solution as it is for solve ROV inference in a non-probabilistic model, as introduced in § V. Gray et al. [14] is the first to bring probabilistic model to binary network tomography, and solves it with MCMC for several scenarios like RFD. However, the drawbacks of MCMC make it hard to generalize to large scale problems. So in this paper, we take a step forward to improve the solution of the inference by introducing SVGD.

XI. Conclusion

In this paper, we propose ROV-MI, a large-scale, accurate and efficient measurement framework for ROV deployment. In the measurement infrastructure, ROV-MI first detects in-the-wild invalid prefixes in the control plane, and the performs traceroutes with probes in the data plane to see which paths are filtering invalid updates, thus labeling the paths. Then in the inference algorithm, ROV-MI models the problem in a probabilisitc setting, associating the deployment of ROV to a random variable distributed in $[0, 1]$. ROV-MI then solves it with an advanced Bayesian inference algorithm SVGD, which combines the particle-based scheme for accuracy and a deterministic descend updating method for efficiency. Our measurement on the Internet shows that ROV-MI can accurately infer the ROV deployment, and the comparison to the baseline methods shows that our algorithm is superior to existing work in terms of accuracy and efficiency.

Acknowledgment

We would like to thank the anonymous reviewers for their valuable comments. We also thank for the suggestions from Han Zhang, Letong Sun, Shize Zhang, Bin Xiong, as well as other members from NMGroup and CNPT-Lab in Tsinghua University. This work was supported in part by National Key R&D Program of China under Grant 2018YFB1800401 and National Natural Science Foundation of China under Grants 62002009. Zhiliang Wang and Xingang Shi are the corresponding authors of this paper.

References

[1] Y. Rekhter K. Lougheed. A Border Gateway Protocol (BGP) . In *RFC 1105*, 1989.

[2] Ratul Mahajan, David Wetherall, and Tom Anderson. Understanding BGP Misconfiguration. 2002.

[3] RIPE NCC. Youtube hijacking: A ripe ncc ris case study. http://www.ripe.net/news/study-youtube-hijacking.html, 2008.

[4] Renesys. China's 18-minute mystery. http://www.renesys.com/blog/2010/11/chinas-18-minute-mystery.shtml, 2010.

[5] Avichai Cohen, Yossi Gilad, Amir Herzberg, and Michael Schapira. Jumpstarting BGP Security with Path-End Validation. In *Proceedings of the 2016 ACM SIGCOMM Conference*. ACM, 2016.

[6] Yang Xiang, Zhiliang Wang, Jianping Wu, Xingang Shi, and Xia Yin. Sign What You Really Care about – Secure BGP AS Paths Efficiently. In *NETWORKING 2012*. 2012.

[7] Ward D. Bellovin S, Bush R. Security Requirements for BGP Path Validation. In *RFC 7353*, 2014.

[8] S. Santesson D. Cooper and et al. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile . In *RFC 1105*, 2008.

[9] Yossi Gilad, Avichai Cohen, Amir Herzberg, Michael Schapira, and Haya Shulman. Are We There Yet? On RPKI's Deployment and Security. San Diego, CA, 2017.

[10] Taejoong Chung, Emile Aben, Tim Bruijnzeels, Balakrishnan Chandrasekaran, David Choffnes, Dave Levin, Bruce M. Maggs, Alan Mislove, Roland van Rijswijk-Deij, John Rula, and Nick Sullivan. RPKI is Coming of Age: A Longitudinal Study of RPKI Deployment and Invalid Route Origins. In *Proceedings of the Internet Measurement Conference*, 2019.

[11] Geoff Huston. Measuring RPKI. In *NANOG 80*, 2020.

[12] Daniele Iamartino, Cristel Pelsser, and Randy Bush. Measuring BGP Route Origin Registration and Validation. In *Passive and Active Measurement*. 2015.

[13] Andreas Reuter, Randy Bush, Ítalo Cunha, Ethan Katz-Bassett, Thomas C. Schmidt, and Matthias Wählisch. Towards a Rigorous Methodology for Measuring Adoption of RPKI Route Validation and Filtering. *ACM SIGCOMM Computer Communication Review*, 2018.

[14] Caitlin Gray, Clemens Mosig, Randy Bush, Cristel Pelsser, Matthew Roughan, Thomas C. Schmidt, and Matthias Wahlisch. BGP Beacons, Network Tomography, and Bayesian Computation to Locate Route Flap Damping. In *Proceedings of the ACM Internet Measurement Conference*, 2020.

[15] Cecilia Testart, Philipp Richter, Alistair King, Alberto Dainotti, and David Clark. To Filter or Not to Filter: Measuring the Benefits of Registering in the RPKI Today. In *Passive and Active Measurement*. Cham, 2020.

[16] Brandon Schlinker, Kyriakos Zarifis, Italo Cunha, Nick Feamster, and Ethan Katz-Bassett. PEERING: An AS for Us. In *Proceedings of the 13th ACM Workshop on Hot Topics in Networks*. ACM.

[17] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Information science and statistics. Springer, New York, 2006.

[18] Qiang Liu and Dilin Wang. Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, 2016.

[19] CISCO. CISCO Route Choice Policy. https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13753-25.html, 2016.

[20] Yossi Gilad, Omar Sagga, and Sharon Goldberg. MaxLength Considered Harmful to the RPKI. In *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies*, 2017.

[21] National Institute of Standards and Technology. NIST RPKI monitoring platform. https://rpki-monitor.antd.nist.gov/Inv, 2021.

[22] Chiara Orsini, Alistair King, Danilo Giordano, Vasileios Giotsas, and Alberto Dainotti. BGPStream: A Software Framework for Live and Historical BGP Data Analysis. In *Proceedings of the 2016 Internet Measurement Conference*, IMC '16, 2016.

[23] RIPE NCC. RIPE Route Information Service. https://www.ripe.net/analyse/internet-measurements, 2021.

[24] University of Oregon. RouteViews Project. http://www.routeviews.org/routeviews/, 2021.

[25] NLnetLabs. Routinator. https://github.com/NLnetLabs/routinator, 2021.

[26] Xingang Shi, Yang Xiang, Zhiliang Wang, Xia Yin, and Jianping Wu. Detecting Prefix Hijackings in the Internet with Argus. In *Proceedings of the 2012 ACM conference on Internet measurement conference - IMC '12*, 2012.

[27] The ZMap Project. https://zmap.io/, 2021.

[28] Oliver Borchert Kotikalapudi Sriram and et al. Origin Validation Policy Considerations for Dropping Invalid Routes . In *RFC draft*, 2021.

[29] RIPE NCC. RIPE Atlas. https://atlas.ripe.net/, 2021.

[30] perfSONAR. https://www.perfsonar.net/, 2021.
[31] Matthew Luckie, Bradley Huffaker, Amogh Dhamdhere, Vasileios Giotsas, and kc claffy. As relationships, customer cones, and validation. In *Proceedings of the 2013 Conference on Internet Measurement Conference*, 2013.
[32] Is BGP safe yet? https://isbgpsafeyet.com/, 2021.
[33] Dootika Vats and Christina Knudson. Revisiting the gelman–rubin diagnostic. *Statistical Science*.
[34] Guanglei Song, Lin He, Zhiliang Wang, Jiahai Yang, Tao Jin, Jieling Liu, and Guo Li. Towards the construction of global ipv6 hitlist and efficient probing of ipv6 address space. In *2020 IEEE/ACM 28th International Symposium on Quality of Service (IWQoS)*, 2020.
[35] Wenjie Xu, Deliang Chang, and Xing Li. On the classification and false alarm of invalid prefixes in rpki based bgp route origin validation. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2019.
[36] Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. *Handbook of markov chain monte carlo*. CRC press, 2011.
[37] Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(5), 2013.
[38] Xin Hu and Z Morley Mao. Accurate Real-time Identification of IP Prefix Hijacking. 2007.
[39] Tomas Hlavacek, Italo Cunha, Yossi Gilad, Amir Herzberg, Ethan Katz-Bassett, Michael Schapira, and Haya Shulman. DISCO: Sidestepping RPKI's Deployment Barriers. In *Proceedings 2020 Network and Distributed System Security Symposium*, 2020.
[40] RPKI Delta Protocol (RRDP) as a future replacement of rsync in RPKI. https://afrinic.net/blog/495-rpki-delta-protocol-rrdp-as-a-future-replacement-of-rsync-in-rpki, 2019.
[41] P. Barford, N. Duffield, A. Ron, and J. Sommers. Network performance anomaly detection and localization. In *IEEE INFOCOM 2009*, pages 1377–1385, 2009.
[42] Alexandros Batsakis, Tanu Malik, and Andreas Terzis. Practical passive lossy link inference. In *Passive and Active Network Measurement*, 2005.
[43] Shinyoung Cho, Rishab Nithyanand, Abbas Razaghpanah, and Phillipa Gill. A churn for the better: Localizing censorship using network-level path churn and network tomography. In *Proceedings of the 13th International Conference on Emerging Networking EXperiments and Technologies*, 2017.
[44] Topology Zoo. http://www.topology-zoo.org/dataset.html, 2021.

APPENDIX A
REASON FOR UNKNOWN ASES

To identify the reason for the *unknown* ASes (i.e., the ASes that we cannot determine whether they adopt ROV or not), we analyze the algorithm from the following two aspects:

First, we investigate whether the **gradient value** in SVGD can give us any clue. For the l-th iteration of SVGD, we define a new variable g^l to represent the sum of the absolute values of the gradient $\nabla_{\theta_k^l} \log p(\theta_k^l|D)$ in Eq. (11) for all the particles:

$$g^l = \Sigma_k \ |\nabla_{\theta_k^l} \log p(\theta_k^l|D)|$$

By this definition, g^l is also a vector of n dimensions (where n is the number of the ASes to infer). Intuitively, when SVGD has not reached convergence, the i-th dimension of g^l indicates **how important AS i is to obtain the observed data (the labeled paths)** D, and small values imply that the observed data D provides low evidence about the ROV deployment of AS i. So we investigate how g^l varies with the number of iteration l and see whether there is a difference between the dimensions of *known* and *unknown* ASes in g^l. The results show that, for the dimensions corresponding to the *known* ASes in g^l, the values are initially relative large, and gradually drops to a small value when l increases. However, for the dimensions of *unknown* ASes, the value remains relatively small even in the initial states. Fig. 13 shows the difference the corresponding dimension of g^l between a *known* AS and a *unknown* AS. This is a indication that the *unknown* ASes are resulted from insufficient evidence in the observed data D.

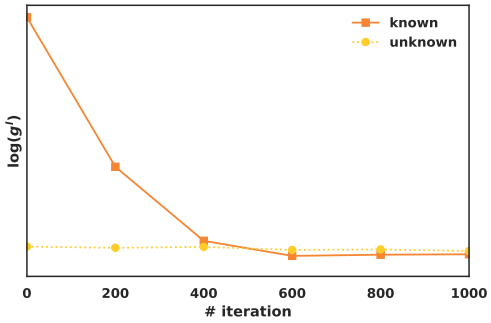


Fig. 13: The variation of the corresponding dimension of g^l of *known* and *unknown* ASes.

Then to confirm our idea, for each *unknown* AS, we collect all the labeled paths in D which contain it to see whether they can provide enough evidence for the inference. Through our investigation, we find the labeled paths of the *unknown* ASes can be divided into two categories.

(1) The first situation can be called **isolated path**, which refers to a path that filters invalid updates, but all the ASes it contains have only appeared very few times in the whole observed data D, and have little overlap with other labeled paths, which makes it hard to infer the ROV deployment of all the ASes on that path.

(2) The second situation, which is more common in the observed data, can be called **masked AS**, which means that in all the labeled paths that contains the *unknown* AS also contains another AS that adopts ROV. In this case, whether the *unknown* AS deploys ROV has no influence on the filtering of these paths, thus **masking** the *unknown* AS.

The analysis of the observed data confirms that the *unknown* ASes are caused by the insufficient evidence in the observed data.

APPENDIX B
EXTENDING TO OTHER SCENARIOS

Below, we introduce how to generalize the inference algorithms. We model ROV deployment inference problem under probabilistic settings, and solve it with SVGD in § V. But in fact, within the field of network researches, there are many other problems that can be modeled and solved in a similar way. A paradigm of this kind of problem is that we want to pinpoint which nodes n have a certain binary property $p(n)$ (e.g., ROV deployment, host failure). However, we usually cannot measure the property of the

TABLE X. Experiments in other scenarios.

Problem	Precision(%)	Recall(%)
RFD measurement	100	86
Link failure localization	100	93

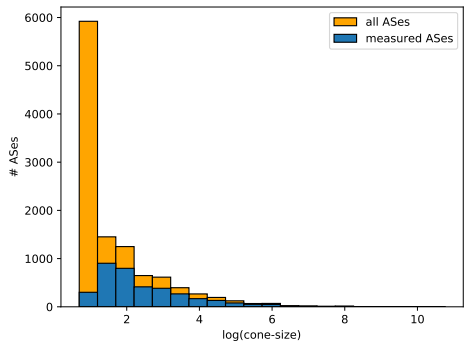


Fig. 14: The distribution of $\log$(cone-size) for all the ASes and all the measured ASes.

nodes directly, instead we can observe whether some paths have the property, and the property of a path d can be expressed as the product of the property of the nodes on it :

$$p(d) = \prod_{n \in d} p(n)$$

This kind of problems are called **binary network tomography**. The key challenges of binary network tomography is how to infer the node properties the observed paths, and our work provides an accurate and efficient solution to it with a probabilistic model and a deterministic Bayesian inference algorithm. We test our algorithm on another two binary network tomography problems, RFD (Route Flap Damping) measurement and link failure localization. RFD is an enhancement mechanism for BGP to prevent frequent anouncement and withdrawal of a prefix, and link failure localizaiton is a common problem in the network. For RFD measurment, we get the observed paths and a subset of ground truth from the measurement by [14], and for link failure localization, we use 12 different topologies in Topology Zoo [44] randomly simulate the failures and vantage points. The results are shown in Table X. Note that there is a slight drop in the recall, which is mainly caused by some missing alarms due to visibility issues. Lacking the visibility for some nodes is the nature of binary network tomography problems and has nothing to do with our inference method. This results illustrate that the inference algorithm of ROV-MI works well as a universal inference method in other binary network tomography problems.

APPENDIX C
CONE-SIZE DISTRIBUTION

Below, we analyze the cone-size distribution of the measured ASes and all the ASes. As the distribution of cone-size is like heavy-tail distribution which is difficult to visualize, we instead investigate the distribution of $\log$(cone-size). Then for all the ASes and all the measured ASes with cone-size larger than 1, we show the binplot of their distribution of $\log$(cone-size) in Fig. 14. It can be seen that the measured ASes contains most of the large ASes (ASes with large cone-size), which indicates that our measurement result is very meaningful for the analysis of the filtering of invalid updates in the Internet.