Measuring the Impact of RPKI on the BGP Updates Volume

SAMUELE QUINZI, Università degli Studi Roma Tre, Italy
CRISTEL PELSSER, UCLouvain, Belgium
GIUSEPPE DI BATTISTA, Università degli Studi Roma Tre, Italy

The Resource Public Key Infrastructure (RPKI) is the primary defense against BGP route misorigination. It offers some barriers against origin hijacks, flavors of route leaks, and misconfigurations. Since its deployment in 2011, the adoption of RPKI by Internet Service Providers has shown continuous growth, a trend that persists to this day. As this growth continues it is important to measure its effect on BGP stability. BGP is a chatty protocol with many updates that can result from a single configuration change on one router. For instance, the addition of a new Route Origin Authorization in the RPKI may result in a change in the validity of a route advertised in BGP, and consequently in different routing decisions that are then propagated in BGP. Here we aim to estimate the volume of updates observed in BGP upon such changes. We identify events resulting from ROA changes and estimate the amount of BGP updates observed in a public BGP repository during these events. We observe that as RPKI adoption rises, the volume of updates generated by RPKI-related changes grows at a similar rate. This growth also mirrors the expansion of the routing table, increasing as more address space is announced and protected by RPKI. However, we also have good news: despite this growth, RPKI-induced updates remain only a very small fraction, corresponding to less than 1% of the total volume of updates.

CCS Concepts: • **Networks** → **Routing protocols**; • **Security and privacy** → **Security protocols**.

Additional Key Words and Phrases: RPKI; RPKI Data Sources; BGP; BGP Updates.

ACM Reference Format:
Samuele Quinzi, Cristel Pelsser, and Giuseppe Di Battista. 2026. Measuring the Impact of RPKI on the BGP Updates Volume. *Proc. ACM Netw.* 4, CoNEXT2, Article 19 (June 2026), 24 pages. https://doi.org/10.1145/3808667

1 Introduction

BGP is the protocol used by Internet routers to exchange routing information and ensure global connectivity to all destinations on the Internet. BGP-speaking routers exchange routes through messages called *BGP updates*. Unfortunately, BGP was not designed with security in mind and it is vulnerable to *hijacks*, where a network advertises a prefix it is not authorized to announce. This may be the result of a misconfiguration or a malicious intent (see, e.g., [5, 21, 33, 38, 42]). The *RPKI* infrastructure is the primary foundation that the Internet relies on today to prevent route hijacking. RPKI consists of a distributed set of repositories that publish *Route Origin Authorizations (ROAs)* and, when necessary, their *revocations*. These objects are issued and signed by *Certification Authorities (CAs)* and define the set of prefixes that each network (*Autonomous System (AS)*) is authorized to advertise.

The effects RPKI has on Internet routing can be described as a three-layer process. First, RPKI repositories are continuously updated with new ROAs and revocations, referred to as *RPKI changes*. Second, caches, also known as *validators*, regularly retrieve these signed objects and verify them

Authors' Contact Information: Samuele Quinzi, samuele.quinzi@uniroma3.it, Università degli Studi Roma Tre, Roma, Italy; Cristel Pelsser, cristel.pelsser@uclouvain.be, UCLouvain, Louvain-La-Nueve, Belgium; Giuseppe Di Battista, giuseppe.dibattista@uniroma3.it, Università degli Studi Roma Tre, Roma, Italy.

This work is licensed under a Creative Commons Attribution 4.0 International License.
© 2026 Copyright held by the owner/author(s).
ACM 2834-5509/2026/6-ART19
https://doi.org/10.1145/3808667

cryptographically. As output, validators produce a set of *authorizations*, each authorizing an AS to announce a certain prefix. A change on the authorization of a prefix (e.g., caused by the revocation of a ROA) constitutes an *RPKI event*. Third, validators distribute the authorizations in their list to routers, which use them to determine the validity state of (prefix, origin–AS) pairs. Routers may behave differently when processing BGP updates that contain *valid*, *invalid* (or whose validity is *unknown*) pairs. For example, a router may (i) discard all announcements containing an invalid pair and accept the others, (ii) accept all announcements with different preferences based on validation states, or (iii) ignore validity altogether. Consequently, an RPKI event may trigger a router to change its routing decisions and, possibly, to send new BGP updates.

Question Q1: Is RPKI harmful for Internet stability? That operators and the IETF worried about exactly this question is not hypothetical: by 2022 the community had already observed that each RPKI event could trigger *route refresh* requests between BGP neighbors, and the IETF issued RFC 9324 [8] specifically to eliminate that behavior, mandating that routers retain paths dropped due to Route Origin Validation in an adjacency table instead of asking their neighbors to resend their RIB. In other words, the standards process had to amend BGP operation because a single ROA change could shake the control plane of many routers at once. Even with this mitigation deployed, a change in validity may still cause a different path selection that then propagates in BGP. At the same time, regulatory pressure is turning RPKI adoption from a voluntary best practice into a near-mandate, driving *large-scale* ROA creation. The U.S. ONCD [25] and the FCC [11], with NTIA's explicit endorsement [23], call on major carriers to create and maintain ROAs covering their address space. This regulatory push comes on top of the growth already visible in the repositories [10, 20, 24, 31, 34] and in the deployment of ROV [3, 18, 28]. The combination of these two trends, *i.e.*, more ROAs to change and more routers acting on every change, raises the question of how the load of RPKI-induced BGP updates will scale. Despite these concerns, *no longitudinal quantification* of the BGP update volume attributable to RPKI events exists to date. Prior work either characterizes RPKI adoption and data quality (*e.g.*, [10, 31]) or analyzes the convergence behavior of individual events [12], but no study has measured, over the full lifetime of RPKI, what fraction of the global BGP update volume is actually attributable to RPKI changes. Closing that gap is the goal of this work. We quantify the impact of RPKI on the global routing infrastructure, *i.e.*, the volume of additional BGP updates exchanged as a consequence of RPKI changes, and track how that volume has evolved as RPKI deployment grew. To identify potential *future trends*, it's essential to analyze how the update load changed *over time*.

Question Q2: Are public RPKI data sources sufficient to reconstruct the history of RPKI events? There are several sources of RPKI data, which differ in key aspects: (1) Some collect ROAs and revocations, others capture RPKI events. (2) Collection frequencies range from seconds to 24 hours. Clearly, when the collection interval is long, the likelihood of losing information increases. (3) Data volumes vary from megabytes to gigabytes per day, affecting processing time. (4) Some cover from RPKI's inception in 2011 and others cover only recent years. Understanding how to use these heterogeneous datasets is essential for both this study and the Internet community.

Question Q3: Can RPKI data and BGP updates be combined to reconstruct the trend of RPKI-driven BGP updates? Measuring updates triggered by RPKI events is challenging. First, we must reconstruct each event's precise timestamp. Second, we need to determine its impact duration. As noted in [12], BGP convergence after an RPKI event can last several hours, during which additional routing or RPKI events may occur, generating further updates.

We establish the necessary terminology in Section 2 and present work related to this paper in Section 7. We discuss available RPKI data sources in Section 3 and develop a method to identify RPKI events from past and current data in Section 4, clarifying to which extent **Q2** can be answered in the positive. Once RPKI events are identified, we address **Q3**, determining how to detect BGP updates

within time frames affected by these events (Section 5). At this point, we have all the components needed to estimate the volume of BGP updates associated with RPKI events versus the total amount of updates. We then describe a straightforward methodology to provide an overestimation of the number of updates related to RPKI events and analyze its evolution alongside overall volume in Section 6. As demonstrated in this section, our overestimation does not weaken the conclusions; although cumulative RPKI-related BGP Updates volume (referred to *RBU-volume* in the rest of the paper) is increasing, it accounts for less than 1% of the total BGP Update volume (denoted *BU-volume*): our answer to **Q1**.

2 Basic Terminology

This section briefly introduces BGP terminology, then explains RPKI terms in more detail. In the examples presented in this paper we use the set of prefixes and ASes proposed for documentation [4].

2.1 BGP basics

The Internet relies on the Border Gateway Protocol (BGP) [27] for inter-domain connectivity. Routers exchange BGP *updates* to convey available paths toward Internet destinations to their neighbors. An update contains, among other information, sets of *announced* prefixes and sets of *withdrawn* prefixes. An *Autonomous System (AS)* is a set of routers and networks managed by the same organization. Each AS is assigned an Autonomous System Number (ASN) by its Regional Internet Registry (RIR), used in BGP to identify routes originated by or passing through the AS. The origin is the rightmost ASN in a BGP update's AS path, while traversed ASes are added to the left.

2.2 RPKI basics

In the RPKI infrastructure, a *Route Origin Authorization (ROA)* [36] is a cryptographic certificate, issued by a *Certification Authority (CA)* and with a *ROA identifier*. It authorizes a certain AS to announce a set of prefixes during a validity period defined by a *start time* and an *expiration time*. The prefixes of a ROA can be grouped into *prefix ranges*, defined by a prefix π and by a *max-length*. A prefix *matches* the range if it is *more specific* than π (including π itself) and its netmask length does not exceed the max-length. As an example, consider the prefix range $192.0.2.0/24 - 25$, where the max-length is 25. Prefix 192.0.2.0/24 matches such a prefix range, while 192.0.2.0/26 does not, as its mask length exceeds 25. Hence, a ROA is the authorization for a certain AS to announce a set of prefix ranges with a start and an end time. Each element of the set is also called *ROA element* and authorizes one range. A *revocation* is used by a CA to invalidate, at a time r, a ROA issued by the same CA. An *RPKI repository* is a set of ROAs and revocations. The revocations emitted by the same CA are grouped into a so-called *revocation list*.

Consider an RPKI repository. An *RPKI change* is the addition of a ROA or of a revocation to the repository. Each change is timestamped, marking either the start of validity of the ROA or revocation time, depending on the case. Given an RPKI repository and a time t, we define *RPKI list* as the set of all triplets $\langle \pi, L_\pi, A \rangle$ such that: (i) there exists at time t a ROA, containing an element with prefix π, max-length L_π, and AS A, whose validity is from s to e. (ii) $s \leq t \leq e$; and (iii) no revocation exists for that ROA at a time $r \leq t$. Such triplets are called *Validated ROA Payloads (VRPs)*. An *RPKI event* is a change in the RPKI list, *i.e.*, the addition or removal of a triplet. Every event stems from an RPKI change, though not all changes produce events. The time of an RPKI event coincides with the time of the RPKI change that generated it. Notice that an RPKI event may affect BGP route propagation.

Three examples follow. Suppose that the RPKI repository contains a single ROA, with id 1, which in turn contains only one ROA element, that authorizes AS 65550 to announce 192.0.2.0/24 with max-length 25 from time s_1 to time e_1. In this case, the RPKI list consists of a single triplet

$\langle 192.0.2.0/24, 25, 65550\rangle$. **(1)** As a first example, suppose that at a certain time $s_2 > s_1$ a new ROA is added to the repository (RPKI change) containing a single element authorizing AS 65551 to announce the same prefix range from time s_2 to time e_2, with $s_1 < s_2 < e_1 < e_2$. In this scenario, three RPKI events occur, at times s_2, e_1, and e_2, respectively. After the first event, the RPKI list contains two triplets: $\langle 192.0.2.0/24, 25, 65550\rangle$ and $\langle 192.0.2.0/24, 25, 65551\rangle$. After the second event, the triplet $\langle 192.0.2.0/24, 25, 65550\rangle$ is removed from the list. Last, after the third event the list becomes empty. **(2)** As a second example, suppose instead that the new ROA contains an element re-authorizing AS 65550 to announce 192.0.2.0/24 with max-length 25 from time s_2 to e_2. In this case, the time of the RPKI change (*i.e.*, s_2) does not coincide with the time of any RPKI event; *i.e.*, the list of triplets remains unchanged at s_2. **(3)** As a third example, suppose that no new ROAs are issued. Instead, a revocation is introduced at time r for ROA with Id 1. In this case, an RPKI event occurs at time r, and at that time the RPKI list becomes empty.

Consider an RPKI event. If it corresponds to the insertion of a new triplet in the RPKI list, we call it an *add-event*, otherwise we call it a *del-event*. The first event of Example **(1)** above is an add-event, while the event of the Example **(3)** is a del-event. Example **(2)** does not contain any RPKI event as there are no changes in the resources declared in the RPKI.

Consider the pair composed by prefix π' and AS A. We say that it is *valid* at time t if, at time t, the RPKI list contains at least one triplet $\langle \pi, L_\pi, A\rangle$ such that π' matches prefix π with max length L_π. Referring to the above Example **(1)**, we have that from s_1 to e_1 the pair 192.0.2.0/25, 65550 is valid. We say that the above pair is *invalid* at time t if no triplet authorizes origin A, while at least one triplet authorizes a different origin. In Example **(1)**, we have that from e_1 to e_2 the pair 192.0.2.0/25, 65550 is invalid. We say that the above pair is *unknown* at time t, if the RPKI list does not contain any triplet $\langle \pi, L_\pi, X\rangle$ (where X is any AS) such that π' matches π with max length L_π. Again, in Example **(1)**, we have that after e_2 the pair 192.0.2.0/25, 65550 and the pair 192.0.2.0/25, 65551 are both unknown.

2.3 RPKI deployment

ROAs and revocation lists (also called *RPKI objects*), are periodically collected by *validators* called *Relying Parties (RPs)*. The validators periodically query all the existing CAs to collect RPKI objects, using protocols like *RRDP* or *rsync* [6]. When queried by a validator at a certain time t, a CA answers providing an RPKI repository, which we call *repository snapshot* (or just *snapshot*), containing the following elements: **(1)** A set of ROAs, such that for each ROA with validity from s to e, it holds that $s < t < e$ and there exists no revocation for that ROA before t; **(2)** A revocation list, containing all the revocations, published before t, for ROAs whose expiration time is after t. We call *global repository snapshot* (or just *global snapshot*) the collection containing all repository snapshots of every CA at a given time. As an example, consider the RPKI repository of a CA containing the ROAs and revocations depicted in Fig. 1. Suppose that three different queries are made to the CA at time t_1, t_2, and t_3, respectively. The snapshot returned at time t_1 contains R_1, R_2, R_5, R_7, R_{13} and R_4 if issued before t_1, in the ROA set, and an empty revocation list. The snapshot at time t_2 contains R_1, R_2, R_3, R_4, R_9, and R_{16} in the ROA set, and ρ_5 in the revocation list. The snapshot at time t_3 contains R_1, R_6, R_{12}, R_{18}, and R_3 if it is not revoked or it does not expire before t_3, and ρ_2 in the revocation list. Note that the revocation list does not include ρ_5 since the expiration time of R_5 is before t_3.

Informally, CAs only distribute ROAs that are valid at the moment the query is issued, and do not distribute revoked ROAs after their revocation time, even if the original expiration time has not yet passed. In contrast, revocations remain listed in the revocation list and are distributed to RPs until the original expiration time of the corresponding ROA is reached.

Relying Parties cryptographically validate the downloaded ROAs and revocations. They compute VRP triplets, generating a local RPKI list. Validators then distribute the VRPs contained in their list

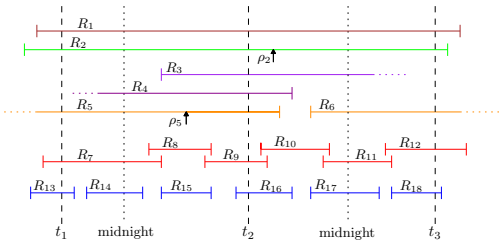


Fig. 1. Timeline with ROAs and revocations. ROAs are shown as horizontal segments, with short vertical ticks marking their start and expiration times. Revocations are depicted with arrows pointing to their occurrence times. ROAs having the same color contain the same AS and the same set of prefix ranges.

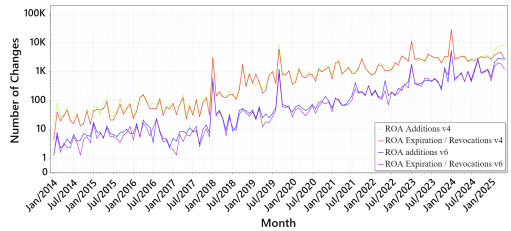


Fig. 2. Number of RPKI changes in the RIPE Archive from 2014 to 2025. Each point on the x-axis corresponds to a month, and the associated y-value represents the total number of changes recorded during the first full week (Mon–Sun) of that month. The y-axis is plotted on a logarithmic scale.

to routers [7], which use them to establish the validity state of (prefix, origin–AS) pairs. Routers may behave differently when processing BGP updates that contain valid, unknown, or invalid pairs. For example, a router may discard all announcements containing an invalid pair, accept all announcements with different preferences based on validation states, or ignore validity altogether. Consequently, any change in the RPKI list (whether an add-event or a del-event) may trigger a router to change its routing decisions and, possibly, to send new BGP updates.

3 Data Sources

In this section we describe the data sources used in this paper. Namely, we first briefly discuss BGP data (Sec. 3.1) and then (Sec. 3.2) we illustrate and compare the RPKI RIPE Archive and the RPKI Flutter data. We also discuss certain timing aspects of Flutter data (Sec. 3.3).

Observe that besides RPKI RIPE Archive and Flutter there are other sources for RPKI data that can be interesting for our purposes. RPKI Views [35] is similar to the RPKI RIPE Archive but with a shorter data availability time period. The NIST Monitor [24] and the MANRS ROA stats tool [20] collect only the validation state transitions of announced prefixes, with collection periods of 6 hours and 24 hours, respectively. However, these long uncertainty intervals make them unsuitable for determining the exact time of an RPKI event.

3.1 BGP Updates Collections

A *Remote Route Collector* (*RRC*) is a BGP speaker that records updates from its peers (*Collector Peers*, *CPs*) tagging each update with its reception time and source CP. Multiple RRCs worldwide collect and store BGP updates, operated mainly by RIPE RIS [30] and RouteViews [40], which include hundreds of CPs. However, BGP traffic volume makes large-scale analysis challenging—for example, a single year of uncompressed updates from one RIPE RIS RRC exceeds 30TB. To keep our analysis tractable, we focus on RIPE RIS RRC00, which is the oldest collector, so it provides the longest history. It is also a very rich collector with 114 CP today, among which 80 CPs share full routing tables. Over the whole measurement period, there were 349 different CPs (see Appendix). The CPs are a balanced mix of IPv4 and IPv6 feeders and they are spread over all the 5 continents. So we believe this diversity reduces the risk of missing out on some local routing dynamics present in the Internet. We acknowledge that other collectors may see a different level of noise.

3.2 Currently Available RPKI-Related Data Sources

3.2.1 RIPE RPKI Archive. The first source is the RPKI Archive (in what follows just *Archive*) maintained by RIPE [29], which has collected daily global snapshots since early 2011 (one snapshot per day). Essentially, the Archive can be regarded as a collection of snapshots of all RPKI objects available at a given time. Snapshots are downloaded to populate the Archive every day, although the interval between consecutive downloads is not precisely fixed at 24 hours. In addition, the time of the snapshots has changed over the course of the acquisition started in 2011. As an example, we observed data captured at 4AM UTC and data captured at 5PM UTC. Due to the snapshot-based collection of the Archive, a ROA appears in all snapshots taken during its validity period, while revoked ROAs are absent from snapshots collected after their revocation time. Instead, revocations first appear in the snapshot collected immediately after the revocation time and remain present in all subsequent snapshots until the original expiration time of the corresponding revoked ROA passes. This implies some redundancy. Also, the Archive contains, for each snapshot, two sets of RPKI objects. One contains the raw data downloaded from the CAs and the other contains only the RPKI objects that pass the cryptographic validation. We consider the second set.

Fig. 2 illustrates the evolution of the global RPKI repository. The plot analyzes the content of Archive snapshots from January 2014 to May 2025, reporting both the number of ROA additions and the number of ROA expirations or revocations, for IPv4 and IPv6. Data are aggregated monthly, with each point on the x-axis representing one month. For each month, we consider the snapshots collected during the first full week (from Monday to Sunday). The y-axis is on a logarithmic scale, so the rough linear growth observed may correspond to exponential growth on a linear scale. Large spikes may indicate large changes such as CA migrations, where all certificates of a CA are revoked and reissued by another CA. This plot reinforces the motivation to study the impact of RPKI events on BGP routing, particularly their contribution to the overall volume of updates in the network.

We use data from 2014 instead of 2011 because early years had very few RPKI changes, with many days showing zero changes [29], making comparisons with later years meaningless.

3.2.2 RPKI-Flutter. The second source is RPKI-Flutter (hereafter *Flutter*) [1, 2], which was deployed on May 1st, 2024. The Flutter collection infrastructure relies on *five* independent, geographically distributed, validators that fetch RPKI objects from CAs with high frequency (each 30 seconds). Based on those objects each Flutter validator computes and records a list of RPKI events. Because of its distributed design, a single RPKI change in the global RPKI repository may generate multiple Flutter events —one for each validator. Moreover, since validators operate independently, different validators may assign different timestamps to events generated by the same change and it is not guaranteed that the effects of an RPKI change is seen by all validators. Fig. 3 compares the number of events collected by Flutter validators from May 1st, 2024 to Aug. 1, 2025. The five curves are quite similar but not identical. Also, the APNIC validator (purple color) started working on Oct. 2024.

3.2.3 Main Differences between Sources. It is important to note that, beyond methodological differences, the two sources collect fundamentally different types of data. The Archive captures full snapshots of the global repository, allowing access to RPKI changes (not to all of them, see the discussion in Sec. 4.2) but not directly to RPKI events, which must be inferred from the set of RPKI objects. In contrast, Flutter exposes RPKI events directly, with no reference to the ROAs or revocations that generated them. Further, the Archive provides precise timestamps for ROA validity start and end, whereas Flutter only records the timestamp when the collecting vantage point observed the change, which may be affected by delays caused by data retrieval and validation. Further comparisons between the two data sources are presented in the next sections.

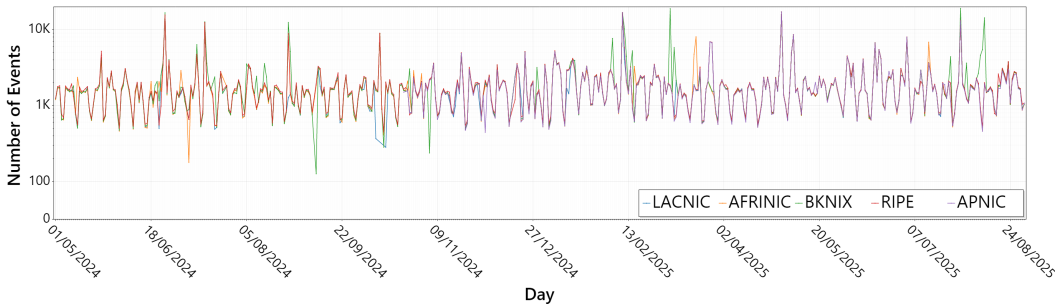


Fig. 3. Number of events collected by the five Flutter validators.

3.3 Realigning Flutter Timestamps using RPKI Beacons

Flutter logs the time as seen by its validators after retrieving and validating RPKI objects, which may introduce a lag between actual RPKI changes and their observation. We estimate this lag to approximate the true timestamp of each RPKI event, aiming to identify the earliest time such changes have some effect on BGP.

Practically speaking, a validator periodically fetches data from a CA, typically operating in so-called "warm mode", where it retrieves only the changes since the previous snapshot rather than downloading the entire repository each time. [32] has shown that validators experience synchronization delays caused by both the network operations required to fetch new data and cryptographic computations. Even in warm mode, these delays can reach the order of minutes and vary depending on the choice of the RP software. To measure the impact of such delays on the Flutter event collection, we performed experiments using a set of RPKI Beacons. The experimental setup is as follows. We announced the prefix 45.132.191.0/24 from AS 3970, and set up a CA to periodically issue and revoke ROAs, thereby changing the validation state of our (prefix, origin–AS) pair. To render our announcement invalid, we issued a ROA authorizing origin AS 0. To make it unknown, we revoked the corresponding ROA. All ROAs where issued using max-length value 24. We further ensured that our experiments were not affected by other routing events. For this purpose we used an additional prefix that remained stable throughout the entire period. We issued a permanent ROA for this prefix, keeping the (prefix, origin–AS) pair consistently valid. We used multiple experimental setups, covering all possible validation state transitions, including the Unknown state. For each experiment, we selected two validation states (Valid-Invalid, Valid-Unknown, and Unknown-Invalid) and generated RPKI events that periodically forced the pair $\langle 45.132.191.0/24, 3970 \rangle$ to transition between them. It is important to note that the triplets $\langle 45.132.191.0/24, 24, 3970 \rangle$ and $\langle 45.132.191.0/24, 24, 0 \rangle$ validating and invalidating our prefix advertisement in BGP correspond to different VRPs.

We fetched from Flutter collectors all events related to the above mentioned VRPs. For all of them we compared their timestamps against the exact timestamps recorded in our logs, relying on the following assumption: the timestamp assigned by a Flutter validator to an RPKI event always occurs after the exact timestamp of the underlying RPKI change. Based on this assumption, for each experimental change we identify a corresponding Flutter event as the one whose timestamp is closest to, but not earlier than, the exact timestamp of the change, regardless of which validator recorded it. Delays are computed as the difference between these two timestamps. Formally, let τ_i be the exact timestamp of change i in an experiment, and let τ_{i+1} be the timestamp of change $i+1$. The corresponding Flutter event is defined as the event with the minimum timestamp τ_f such that

$\tau_i \leq \tau_f < \tau_{i+1}$. The delay is computed as $\tau_f - \tau_i$. If no Flutter event is observed in $[\tau_i, \tau_{i+1})$, we mark the change as unmatched. In our experiments, all changes successfully matched a Flutter event. We have that more than 95% of the delays we observed were below 5 minutes, in line with [32]. The CDF of the delays computed for all changes performed during our experiments is in the Appendix.

As an additional validation step, for each experimental change we analyzed BGP traffic observed by 79 CPs. We ensured routing convergence by waiting a sufficient amount of time before triggering the next event. In [12], the authors measured convergence delays for RPKI events using a beaconing methodology similar to the one used in this work. We leveraged their results to determine an appropriate waiting period. Then, for each experiment event i, we measured the number of BGP updates starting from three different timestamps: (1) starting from the exact timestamp τ_i for the duration of the convergence time; (2) starting from the Flutter timestamp τ_f until τ_{i+1}; and (3) starting from the Flutter timestamp shifted backward by five minutes (i.e., $\tau_f - 5$ min) until τ_{i+1}.

We found that using $\tau_f - 5$ min is an excellent estimation of the exact starting time of RPKI events. Indeed, the number of updates in case (3) coincided in 99.9% of the instances with the numbers observed for case (1), which is our ground truth. Our validation confirms the works of Sediqi et al. [32] and Fontugne et al. [12]. In the subsequent analysis we therefore shifted all Flutter event timestamps 5 minutes backward before measuring RBU-volume.

4 Setting Up a Time Machine Using the RIPE Archive

As discussed in Section 3, the Flutter dataset is ideal for analyzing historical RPKI events and their impact, but it spans only the last few months. We therefore introduce a method to compute RPKI events from the Archive dataset (Section 4.2) and validate it against Flutter data (Section 4.3), considered as the ground truth for the period when the two data sources are both available. Before this, we address the challenge posed by the prohibitive data volume (Section 4.1).

4.1 A Reference Set of Days for Data Analysis

To reduce the amount of Archive data and BGP updates to process, we sampled one day per week across the entire period. To choose the weekday to focus on, we analyzed the distribution of RPKI events by day in the first full week of each month (Monday–Sunday), as detailed in Sec. 3.2.1. Fig. 4 shows the resulting distribution of RPKI changes by weekday. The distributions for Monday and Tuesday stand out, reaching higher values on the x-axis and showing the presence of more bins with larger ROA change counts. In the rest of the paper, we choose to focus on Monday rather than Tuesday because its distribution reaches a higher maximum number of events, thus providing an upper bound for RBU-volume estimation. In what follows, all analyses are based on data from all the Mondays between Jan. 2014 and Aug. 2025.

4.2 From Archive Snapshots to RPKI Events

We have seen in Sec. 3 that for an analysis of the evolution of RPKI we have two data sources: Flutter that is reliable and complete and Archive which is not complete and that provides only RPKI changes but not RPKI events. Unfortunately, the first has a very small time span. Hence, for analyzing the period from 2014 to April 2024 we had to rely only on the Archive.

To do that we computed, for that period, the RPKI events from the Archive, with the methodology described below. Let us focus on a specific day d of interest. First, we collected from the Archive all the available RPKI objects that may affect the events during d. Second, we extracted from such objects all the ROA elements, with their validity periods, given by the issuing timestamps of ROAs and the expiration or revocation timestamps. Third, we computed a timeline of all the insertions and deletion of a VRP that happened during d, and associated an add-event to each insertion and a del-event to each removal. In the following we describe the main aspects of this methodology.

4.2.1 Three Snapshots are (Almost) Enough. An exhaustive list of the Types of RPKI objects that may affect events during d is the following (*e.g.*, see Fig. 1 where d is limited by its surrounding midnights): ROAs with starting time before d and: (1) that expire or are revoked after d (*e.g.*, R_1), or (2) that expire during d (*e.g.*, R_4, R_7, and R_{14}), or (3) that are revoked during d (*e.g.*, R_2 and R_5). ROAs with starting time during d and: (4) that expire or are revoked during d (*e.g.*, R_8, R_9, R_{10}, R_{15}, and R_{16}) or (5) that expire or are revoked after d (*e.g.*, R_3, R_6, R_{11}, and R_{17}).

While it is clear why ROAs of Type (2)–(5) may cause events during d, it is more subtile the fact that ROAs of Type (1) can also influence events. To understand this, consider, as an example, Fig. 1. Suppose that R_1 contains the ROA element with prefix 192.0.2.0/24, max-length 24, origin AS 65550. At beginning of day d, the VRP $\langle 192.0.2.0/24, 24, 65550 \rangle$ is present in the RPKI list. Suppose that a new ROA R_k is issued during d, containing only a ROA element authorizing the same origin AS to announce the same prefix range. The VRP corresponding to this element is again the triplet $\langle 192.0.2.0/24, 24, 65550 \rangle$, which is already present in the RPKI list. Hence, the change given by the addition of ROA R_k doesn't generate an event due to the presence of R_1. Therefore, we need to collect ROAs of Type (1) to exclude these cases from the list of inferred events.

To capture the RPKI objects described above, we retrieve from the Archive the snapshot of day d and those of days $d - 1$ and $d + 1$. These snapshots contain all the ROAs and the revocations that the Archive can provide and that can affect the RPKI events happening during d. In the Appendix, we show that such 3 snapshots are necessary and sufficient for our purposes.

4.2.2 Missed RPKI objects. Note that short-lived ROAs may be missing from the dataset. We define a ROA as *short-lived* if its validity duration is less than 24 hours, *i.e.*, less than the amount of time needed to be surely captured by at least one snapshot. Precisely, a ROA issued after a certain snapshot and revoked (or expired) before the next snapshot will not appear in any snapshot and is not accounted for. Fig. 1 shows many examples of ROAs that are missed by the Archive, in particular R_8, R_{10}, R_{11}, R_{14}, R_{15} and R_{17} are all missed ROAs. Instead, R_9, R_{12}, R_{13}, R_{16} and R_{18}, despite being short-lived ROAs, are captured by the Archive since their validity period includes at least one snapshot time. Note that revocations may be lost too. In fact, if the original validity period of a

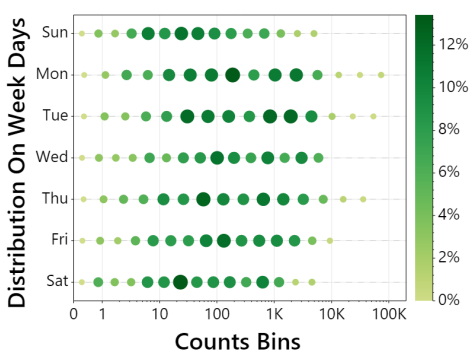


Fig. 4. Distribution of ROA change counts by weekday. For each day, data from the first week of every month in the analyzed period is aggregated. Green dots indicate probability density, with color intensity and size encoding probability: darker and larger dots represent higher values.

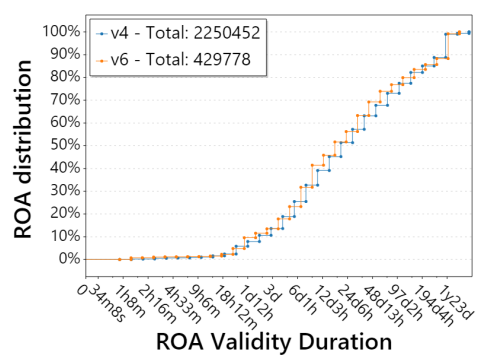


Fig. 5. CDF of ROA validity durations, limited to ROAs that were valid during the analyzed days (*i.e.*, those starting on, ending on, or spanning a Monday). The x-axis is in log scale. The legend reports the total number of ROAs considered for each prefix family.

revoked ROA doesn't include the first snapshot time after the revocation, then the revocation of the ROA would not appear in any snapshot.

The presence of short-lived ROAs is confirmed by Fig. 5, which shows the CDF of the validity period duration of all ROAs considered in our analysis, *i.e.*, those whose validity period overlaps with at least one Monday. A fraction of short-lived ROAs is visible, indicating that such ROAs may exist in the global repository but are difficult for the Archive to capture. Indeed, the CDF begins to rise around 12 hours and shows its first significant increase between 24 and 36 hours, corresponding to ROAs whose validity period covers at least one snapshot.

4.2.3 Estimating RPKI events. After collecting all available objects, we analyze VRP insertions and deletions in the RPKI list to estimate events. This estimation may be imprecise due to missing objects; short-lived ROAs, for instance, can cause both underestimation and overestimation of events. Refer to Fig. 1 and focus on the red and blue ROAs. ROAs of the same color correspond to the same origin AS and include the same set of prefix ranges.

The first case (*Underestimation*) is straightforward: if a ROA is absent from the dataset, any RPKI event it generates is also missing. Examples of this case are given by the blue ROAs. For each of them, its issuance involves the addition of the corresponding VRPs to the list, and its expiration involves the removal of such VRPs, hence generating events. Since R_{14}, R_{15} and R_{17} are missed by the Archive, it is impossible to detect the corresponding events.

The second case (*Overestimation*) is more subtile. Consider the red ROAs in the picture. Because each ROA issuance or expiration overlaps with another ROA (except for R_7 issuance and R_{12} expiration) containing the same elements, none of them generates an event. But, unfortunately, R_8 is missing from the Archive. Hence, when building the timeline for the RPKI list, we remove the VRPs corresponding to the elements in R_7 when it expires, generating del-events. Also, we add such VRPs again to the list when R_9 is issued, generating the corresponding add-events. Further, since a ROA contains a set of prefix ranges, and each of them corresponds to a different VRP, this overestimation may be amplified: missing a single ROA could generate many false events.

Now we compare the events we inferred from Archive against the events collected by Flutter.

4.3 Events Obtained from the RIPE Archive vs Flutter Events

To assess the effectiveness of our usage of Archive data for computing RPKI events, we compared the events derived from the Archive between May 2024 and August 2025 with those obtained from Flutter in the same period. As shown in Fig. 6, the two curves are largely consistent, with a noticeable overestimation in the most recent months. Since the five Flutter validators do not necessarily observe the same number of events (Fig. 3), in the figure we report their average.

However, RPKI events are only a part of the story. Namely, when we have to compute the burden originated by events, we have to consider the number of involved prefixes and whether they are actually announced in BGP. Fig. 7 compares the number of prefix ranges observed in all events obtained from Archive data and those recorded in Flutter data, for each analyzed day. We have that the number of prefix ranges observed from Archive presents an overestimation that is similar to the one we have for events. The discrepancy in the number of prefix ranges hints the presence of a certain behavior of some CAs. In fact, the Archive observing more prefix ranges indicates that there are CAs that issue and revoke ROAs following a pattern similar to the one represented by the red ROAs in Fig 1. Indeed, from such a pattern, Flutter does not capture any events for the corresponding VRPs, and the affected prefixes are therefore absent from its data. On the contrary, due to missing ROAs, the Archive data may lead us to infer the presence of events for those same VRPs. Moreover, due to the amplification effect that occurs when a single ROA covers multiple prefix ranges, the loss of only a few ROAs can result in a larger number of prefixes appearing

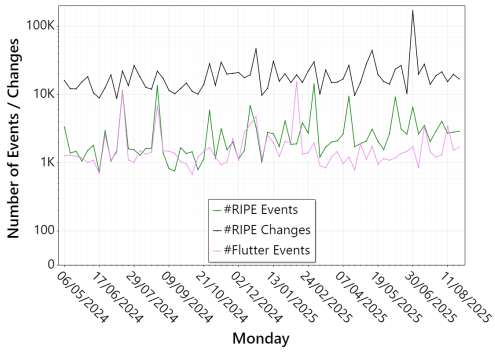


Fig. 6. Number of RPKI changes in the Archive, number of RPKI Events from Flutter, and number of RPKI events computed using the Archive.

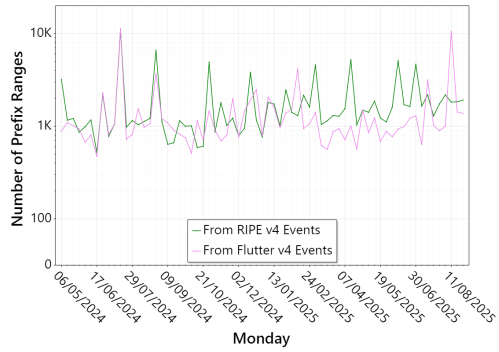


Fig. 7. Number of prefix ranges observed either using the Archive or using Flutter data.

to be affected by RPKI events, even though they were not. As shown in Fig. 5, the proportion of short-lived ROAs is relatively small; consequently, the number of ROAs we miss is also limited.

Since it is impossible to determine whether an event inferred from the Archive is a false event in historical data, we can only acknowledge that our measurement of RBU-volume for historical periods likely includes background fluctuations, and should therefore be regarded as an overestimation.

5 A Methodology for Measuring RPKI Originated BGP Updates

In this section, we show a method to measure the BGP updates generated by the RPKI events identified in Sec. 4. This requires addressing three challenges: determine the best possible approximation of the timestamp at which an event occurred (Sec. 5.1); determining a suitable time threshold that captures the BGP convergence time following an RPKI event (Sec. 5.2); and clustering RPKI events that affect the same prefix and whose impact on BGP routing may overlap in time (Sec. 5.3).

5.1 Flutter Validators and Exact Event Timestamps

As described in Sec. 3.2, Flutter data collection relies on multiple validators distributed worldwide, each operating independently. As a consequence, the same RPKI event may be seen multiple times, with each copy potentially carrying a different timestamp. Since Flutter does not provide explicit references to the underlying RPKI changes (*e.g.*, which specific ROA issuance or revocation triggered the event), identifying duplicates of the same event across different validators is non-trivial—particularly when many events occur in close temporal proximity. Therefore, the problem of estimating the exact timestamp of an RPKI event, and determining the correct point in time from which RBU-volume should be measured, has to be addressed. One could argue that it is sufficient to focus only on a single Flutter validator. However, this approach presents two significant shortcomings. First, validators fetch from CAs with variable jitter around their synchronization interval. Consequently a validator may miss events, corresponding to prefixes that are only registered for a short time, that are nevertheless captured by other validators. Second, even if all validators detect the same set of events, they may record them in different orders due to the varying intervals. As discussed in Sec. 3.3, we operate under the assumption that Flutter event timestamps always *follow* the exact time of the underlying RPKI change. Consequently, we are interested in retaining the earliest available copy of each RPKI event across validators. For both reasons, increasing the number of validators improves the precision of the dataset. There is a higher likelihood to catch all events and closer to their appearance in the RPKI. In Sec. 5.3, we introduce

a clustering methodology that overcomes these inconveniences. Combined with the previously described 5-minute time-shifting adjustment for Flutter events, this method provides a reliable foundation for determining the timestamps used to measure BGP updates.

5.2 Selecting a Convergence Time Threshold

As shown in [12], the BGP convergence time following an RPKI event that changes the validation state of a (prefix, origin–AS) pair is significantly longer than the convergence time observed after standard BGP events (e.g., prefix announcements or withdrawals). This difference arises from the heterogeneous synchronization delays between RPs and CAs worldwide. The authors report that such convergence delays can reach a median value of nearly one hour. On our side, we need to estimate a convergence time threshold that is conservative enough to both including all the updates related to RPKI, and avoid unrelated updates in our measurements of RBU-volume. Hence, beyond relying on the results of prior work, we also measured convergence time using the RPKI beacon experiments illustrated in Sec. 3. We estimated the convergence time for each CP cited in Sec. 3.1 as follows. Let t_i denote the timestamp of an experiment event i and t_{i+1} the timestamp of the following event. The convergence time for event i is measured as the difference between the timestamp of the last BGP update observed within the interval $[t_i, t_{i+1})$.

We have observed that convergence was always reached within three hours. This finding is consistent with the results of [12], supporting the choice of three hours as a reasonable threshold. We therefore adopt it as a conservative trade-off: large enough to capture BGP updates related to RPKI events, while minimizing the inclusion of unrelated updates. A figure in the Appendix shows the CDF of the estimated convergence times measured during our RPKI beacons experiments.

5.3 Clustering RPKI Events

As discussed earlier, when measuring BGP updates induced by RPKI events, it is essential to account for events whose impact on routing may overlap. An example is provided by the events collected by Flutter: for each event, multiple occurrences may exist, each recorded by a different validator, with timestamps that can differ due to different synchronization delays and processing latencies (see Sec. 3.2). As another example, consider a prefix range, and denote it with the pair $\langle \pi, L \rangle$ and a distinct prefix $\pi' \neq \pi$ such that π' matches $\langle \pi, L \rangle$, and suppose π' is announced. Suppose two RPKI add-events occur for the VRPs $\langle \pi, L, A_1 \rangle$ and $\langle \pi', L', A_2 \rangle$. Both events involve prefix π'. The first event affects the validity of the announcement of π' from origin A_1, changing it from Unknown to Valid, while after the second event also the announcement of π' from A_2 becomes valid. If these events occur close in time, the BGP updates observed after the second event may in fact have been triggered by the first. If such events were considered independently, the measurement would result to a significant overestimation of RBU-volume, since most updates would be counted twice. To address this, we adopt a one-dimensional clustering technique that groups RPKI events occurring close enough in time to interfere with each other in terms of their impact on prefix routing. For each cluster, we then identify the overall time interval that is relevant for measuring the related RBU-volume. The clustering works as follow. Consider a day d, and let E_d be the set of all RPKI events collected on d. Each event $\epsilon_i \in E_d$ refers to a triplet $\langle \pi_i, L_i, A_i \rangle$. Now, consider a prefix π for which we observe BGP updates during d, and let τ_{conv} denote the convergence time threshold selected in Sec. 5.2. We proceed as follows. First, we extract from E_d all events ϵ_i such that π matches the prefix range $\langle \pi_i, L_i \rangle$, *i.e.*, all events that may affect the routing of prefix π. We then construct a timeline by ordering these events by their timestamp. Starting from the first event ϵ_i in the timeline, we build a *cluster* c according to the following procedure: (i) Let $I_c = [t_i, t_i + \tau_{conv})$ be the *current interval*, where t_i is the timestamp of ϵ_i. (ii) Collect all events ϵ_j with timestamp $t_j \in I_c$ and add them to the cluster. (iii) Update the current event ϵ_i to be the last event in the

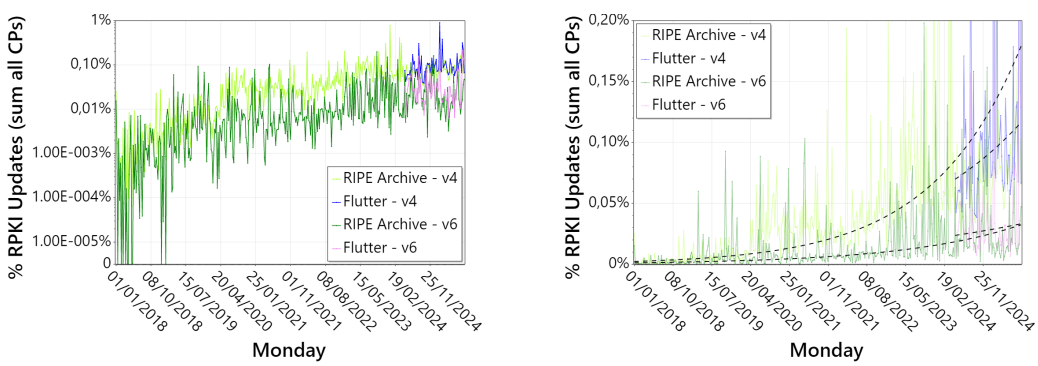
cluster, in chronological order. (iv) Repeat the process until no additional events fall within the current interval. At this point, cluster c is formed. Let ϵ_s and ϵ_e be the first and last events in c, with timestamps t_s and t_e, respectively. The *relevant interval* of the cluster for measuring RPKI-induced BGP updates is defined as $[t_s, t_e + \tau_{conv})$. If events remain unclustered after timestamp $t_e + \tau_{conv}$, a new cluster is initialized starting from the first of such events in chronological order. Finally, all BGP updates for prefix π that fall within at least one of the cluster intervals are considered RPKI-related updates. This clustering technique solves also the problem of multiple copies of the same event in the Flutter data, since the first event in a cluster will always correspond to the earliest among its copies. Furthermore, when multiple events occur close in time and their copies interleave, the clustering process ensures that all such events are grouped into the same cluster.

6 RPKI is not Harmful for BGP Stability

In this section, we examine whether RPKI-related BGP updates currently pose, or could potentially pose in the future, a threat to BGP stability. Our analysis is based on the datasets and methodologies described in the previous sections. The big picture of our results is in Fig. 8.

6.1 RPKI-related BGP Updates Volume vs Total BGP Updates Volume

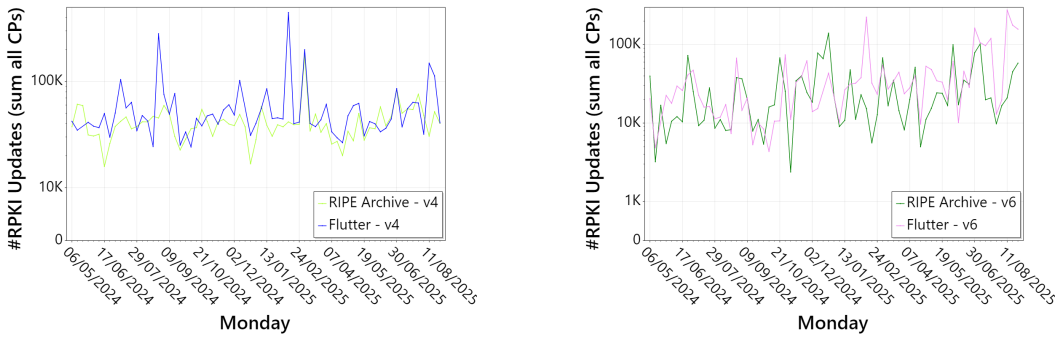
Figures 8a–b show the timeline of the percentage of RBU-volume relative to the total BU-volume collected across all analyzed days. For readability, the plots display only the period from January 2018 to August 2025, as values prior to 2018 contain multiple zeros and 2018 marks the first year of noticeable RPKI adoption growth [24]. A figure covering the entire period from 2014 is provided in the Appendix. Additionally, on May 20, 2024, and Dec. 16, 2024, Flutter data were missing; these days were excluded from the analysis, and the plots show an interpolation between the preceding and following analyzed days. The figures lead to two main observations. First, they point out that the fraction of RBU-volume was and currently is almost negligible compared to the total number of updates, remaining below 1% of the daily update volume, even in recent years and even when there are peaks of RPKI events (see Fig. 3). Consider that, as discussed in Sec. 4.3, the RIPE Archive events are an overestimation of those observed by Flutter. Also, the amounts of updates computed starting both from Flutter events and from Archive events are an overestimation, since we count all the updates in the convergence interval of RPKI events. Indeed, we are possibly capturing updates that



(a) Timeline showing the percentage of RPKI-related updates over the total amount of updates for the analyzed period (log-scale y).

(b) Version of Fig. 8a using a linear scale on the y-axis. Peaks above 0.2% are removed for readability. The figure includes 4 exponential regression curves.

Fig. 8. Timelines illustrating the increase in the percentage of RPKI-related updates.



(a) IPv4 (b) IPv6

Fig. 9. Comparison between the number of RPKI-related updates observed using Flutter Events and using RIPE Archive Events in the period from May 2024 to Aug 2025.

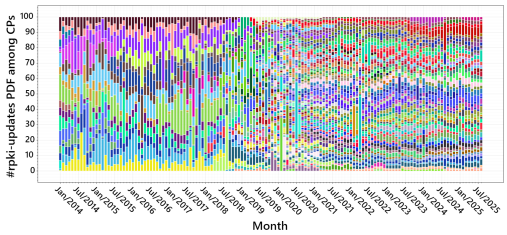
are related to other routing events. Second, although the percentage remains low, a clear upward trend is evident. Note that the y-axis in Fig. 8a uses a log. scale; therefore, the approximately linear trend corresponds to an exponential growth (with a small exponent) when represented on a linear scale. Fig. 8b further illustrates this point. Here, the y-axis is limited to 0.2%, truncating spikes to improve readability, while the four dashed black lines represent the regression curves fitted to the respective datasets. The two regressions spanning the entire period correspond to Archive data, whereas those shorter correspond to Flutter data. The two upper curves represent IPv4, while the lower ones correspond to IPv6. The Flutter and RIPE curves exhibit similar trends, and both indicate that the percentage has approximately doubled over the past year. More precisely, the percentage of the RBU volume, measured using Flutter events, was 0.069% (IPv4) and 0.024% (IPv6) in May 2024, and increased to 0.116% (IPv4) and 0.033% (IPv6) in August 2025.

To facilitate a comparison between the RBU-volume observed using Flutter events and that obtained from Archive-based events, we present Fig. 9. This figure provides a detailed view of the updates volume during the period when Flutter data was available, showing curves for IPv4 (a) and IPv6 (b). The comparison reveals that the resulting volume levels are quite similar, despite the overestimation in the number of events. This similarity also helps interpret the regression curves in Fig. 8b. Specifically, since the regression curve for Archive lies above that for Flutter and the recent volume levels are comparable, the difference between the two regressions can be attributed to the historical analysis, where Archive data shows a steeper increase in the percentage of updates.

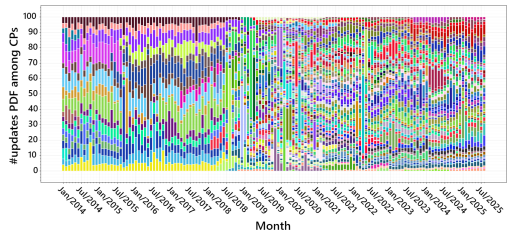
From now on we present results that combine both data sources, relying on Archive for historical analysis and switching to Flutter events from May 2024.

6.2 Distribution of RPKI-related BGP Updates Volume among CPs

A question is whether the RBU-volume has been evenly distributed among CPs over time. The data represented in Fig. 10a answer this question in the positive. The figure tells the story of how the perception of the RBU-volume (IPv4 only) has been distributed among CPs. Each x-value corresponds to one month. The sequence of segments above x is a stacked bar plot where each bar is associated with a CP. The height of a bar shows the percentage of RPKI originated BGP updates collected by a CP during month x. Each CP is assigned a color which is the same across time. Also, from bottom to top the CPs are presented in the same order in all the stacked bars. If at month x a certain CP recorded 0 updates, then its bar in the corresponding stacked bar is omitted. Fig. 10b has



(a) Distribution over time of the number of RPKI-related updates among CPs.

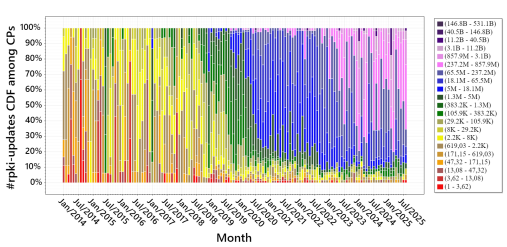


(b) Distribution over time of the total number of BGP updates among CPs.

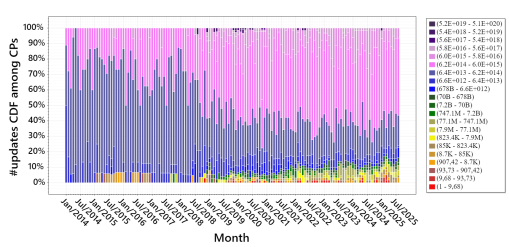
Fig. 10. Comparison between the distribution over time of the RBU (a) and the BU-volume (b) among CPs.

the same structure. Similarly to the previous one, it depicts the story of the distribution of the *total* number of BGP updates collected by CPs with the same representation, and if the same CP appears in both plots, it is associated with the same color. Since the second plot refers to all the updates collected, a CPs doesn't appear on the plot in month x if it didn't collect any update at all during x, *i.e.*, the CP was probably inactive or not yet deployed. As a collateral effect, the figure shows the evolution of the number of CPs at RRC00. This number increased over time, which explains why the height of the bars tends to decrease in both plots. From the comparison of the two figures, we conclude that the RBU-volume has been rather uniformly distributed during the years.

While the evolution of the percentage of RBU-volume suggests that the impact of RPKI on routing stability is negligible, it hides the actual volume of updates related to RPKI, its evolution over time, and how it compares with the total number of observed updates. Fig. 11a and 11b aim to highlight this. The figures show the evolution of the CDFs of updates volumes (IPv4) collected across CPs over the entire analyzed period, for RBU-volume and BU-volume, respectively. Similarly to figure 10, each point on the x-axis corresponds to one month. A stacked bar plot above one x-value represents a CDF, where each colored bar is associated with a bin of values. The bin sizes increase exponentially from bottom to top. The colors are associated with the bin ordering, and the corresponding values are mapped in the legend to the right of the plots. Red and orange bars are associated with bins containing smaller values; yellow, green and blue indicate the presence of intermediate values, and violet and indigo are associated with the highest values. The height of a bar indicates the percentage associated with the corresponding bin. In both plots, bin sizes are computed based on the maximum value observed across the full period of the plot, ensuring comparability over time. As an example, consider the rightmost bar of figure 11a, which depicts

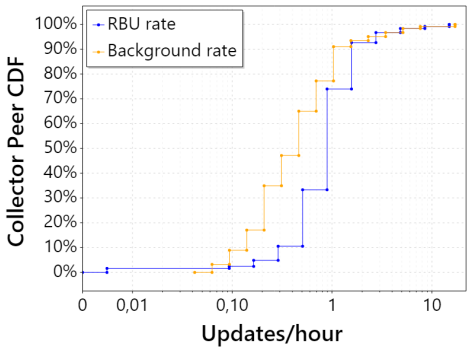


(a) CDFs over time of the number of RPKI-related updates collected by CPs.

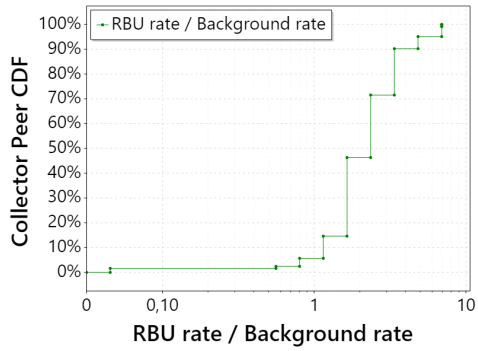


(b) CDFs over time of the total number of updates collected by CPs.

Fig. 11. Comparison between the RPKI-related updates (a) and the total BGP updates (b) collected by CPs.



(a) CDFs of the update rates recorded by CPs. Blue: RPKI-related BGP Updates. Orange: Not RPKI-related BGP Updates.



(b) CDFs of the ratio between the BGP update rate during RPKI events and the BGP update rate outside RPKI events.

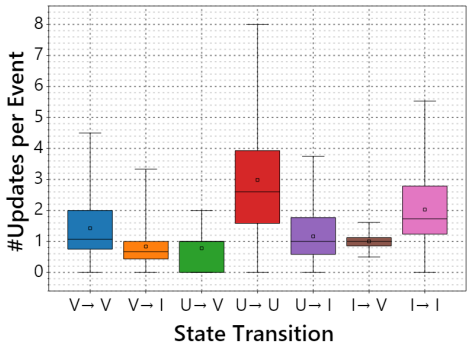
Fig. 12. between BGP update rates. Flutter events.

the CDF of the number of RPKI related BGP updates on August 2025. From bottom to top, it shows that: **(i)** a very small percentages of CPs collected few RPKI related updates (the small red and orange bars); **(ii)** no CPs collected a number of updates between $2.2K$ and $105.9K$ (absence of any yellow bar); **(iii)** around 30% of CPs fall into the green and blue bars (the total height of the bars goes from 5% to 35%); **(iv)** most of the collectors fall into violet bars, collecting from hundreds of millions to few billions of updates, and **(v)** a very small percentage (around 2%) of CPs collected tens of billions of RPKI related updates, represented by the darker indigo bin on top of the stack.

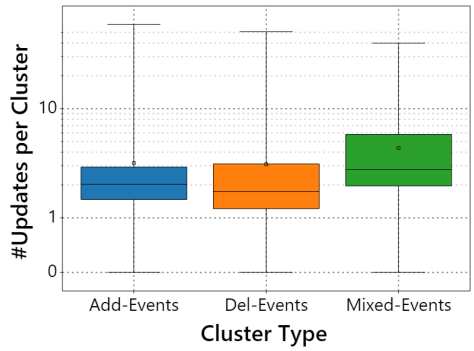
Looking at the whole story, on the left side of the picture, red, orange, and yellow bars dominate, reflecting a low number of updates. This isn't surprising, since on its early years, RPKI had a low level of adoption. The emergence of green and blue bars marks a sharp growth in the number of updates, a growth that follows the increased adoption of RPKI to protect announced prefixes [24], as well as the increase of the global routing table [16]. Towards the right side, indigo and violet bars appear, indicating that the number of RPKI related updates steadily increased in recent years. Fig. 11b shows the evolution of the CDFs of the BU-volume collected over time, and makes it possible to compare the scale of the two update volumes. The color and bar convention is the same as in the previous figure. However, the ranges are different: values grow rapidly into the billions of updates, and eventually into the order of millions of trillions. The absolute dominance of blue and violet bars indicates that most CPs collected very large volumes of updates. A growth in the overall number of updates is still visible, highlighted by the darker violet bars emerging on the rightmost side of the plot. It is precisely this huge scale difference that highlights how, even after the abrupt increase we see in Fig 11a, the RBU-volume remains only a marginal contributor to the overall causes of routing updates.

6.3 Comparison with Background Noise

We also analyzed how the frequency of BGP updates varies between time intervals containing RPKI-related updates and what can be considered background BGP noise. To this end, on the same days as in the previous analysis, for each prefix affected by an RPKI event, we identified the corresponding RPKI event clusters. We then divided time into two types of intervals: those corresponding to the clusters (relevant intervals) and the remaining periods. We computed the BGP update rates for both types of intervals and finally aggregated these rates across all prefixes



(a) Updates per state transition. The states are V = Valid, I = Invalid, and U = Unknown.



(b) Updates per cluster type. The y-axis is log scale.

Fig. 13. Box plot showing the distribution, across CPs, of RPKI-related BGP updates. The median, the 25th, and 75th percentiles are represented by the boxes, while the average is shown as a small black square. Minimum and maximum values are indicated by the whiskers. Flutter Data.

observed by each CP. Fig. 12 illustrates the result of the analysis restricted to Flutter events (figures on the Archive events are in the Appendix). Namely, Fig. 12a shows the CDF of both the update rates. As expected, the curve of the RPKI-related updates is to the right of the curve of the background BGP noise, meaning that the update rate increases during an RPKI event. Fig. 12b allows to compare the two rates. It shows that in 90% of cases the ratio is at most 4 and that in about 50% of cases such a ratio is lower than 2. While the ratio is rather high the update rates remain low, below one packet per hour in most cases, in RPKI-event time windows.

6.4 State-transition Analysis

Another interesting question is whether certain types of state transitions cause more updates than others. As discussed in Sec. 5.3 the effects, in terms of updates, of events that are close in time are barely distinguishable. Hence, to investigate this aspect we focused on the global effect of each cluster, considering the validation state of a prefix-origin pair before the cluster and after the cluster. We then aggregated per CP. Since the number of events per transition type observed by a CP is very different, we normalize the number of updates per CP by the total number of events that lead to the state transition. Fig. 13a illustrates the result of this analysis restricted to Flutter events (figures on the Archive events are in the Appendix). We observe that the average is quite low and very similar for every type of transition. Transitions from Unknown to Unknown create more noise, as well as transitions from Invalid to Invalid.

6.5 Event-Type Analysis

In Sec. 2.2, we defined the concepts of add-events and del-events. It is of interest to determine whether one event type generates more RPKI-related BGP updates than the other. For the same reasons as discussed in Sec. 6.4, we conduct the analysis at the cluster level. Specifically, we classify clusters into three categories: clusters containing only add-events (*Add-Event clusters*), clusters containing only del-events (*Del-Event clusters*), and clusters containing both types (*Mixed-Event clusters*). The distribution of the number of updates per cluster is shown in Fig. 13b. It shows that, in terms of contribution to BGP updates, Add-Event and Del-Event clusters exhibit quite similar overall values while Mixed-event clusters exhibit an slightly higher number of updates on average.

6.6 Discussion

We identify three key factors that likely shape the impact of RPKI: the scope of ROA address space coverage, the scale of ROV deployment, and the frequency of RPKI events. Current evidence suggests that: ROA coverage is both reasonable and expanding [24]; ROV adoption remains limited [18]. Given that RPKI events are not uncommon (see Fig. 3), with Flutter collectors recording around $1K$ events daily, we posit that the primary explanation for the relatively low RPKI noise is that ROV deployment is still in its infancy. Establishing a methodology to measure this noise now is crucial, as it will enable us to track its evolution as adoption grows. Therefore, we can conclude that the observed increase in percentage is better interpreted as a natural consequence of the progressive adoption of RPKI by Autonomous Systems on the Internet, rather than a threatening challenge to BGP stability. At least, for now.

7 Related Work

Among prior works in studying the RPKI infrastructure and retrieval of objects, Kristoff et al. focused on how often validators retrieve RPKI data from publication points [17]. They observe that 20% of validators fetch less than 20 times per day and conclude the fetching process can be incomplete. Validators may possess outdated data leading to potentially erroneous routing. [41] and [22] have analyzed the validator's software. They discovered several bugs at the root of security vulnerabilities. Researchers have also inferred the deployment of Route Origin Validation in the Internet from active measurements [9, 13, 18, 39]. [19] studies RPKI in conjunction with Remotely-Triggered Black Holing (RTBH) to understand their combined uses as DoS mitigation. Deployments of ROV does not have the same coverage breadth as shown in [14]. [43] studies the coverage of routes for popular Web destinations. Huston [15] provides a view of the amount of prefixes advertised per AS and the update rate per second on May 8, 2025 for a feed collected at APNIC R&D, and observed several peaks above 150 updates/s. Stucchi [37] studies BGP updates with communities indicating the validity of the route's origin. These communities are tagged with two ASs, and are present in IPv4 and IPv6 except for the community indicating the origin is invalid.

8 Conclusions and Future Work

To answer Question **Q1** of the Introduction, we analyzed 11 years of RPKI deployment, measuring the impact of RPKI events on routing stability. We developed a methodology (addressing **Q2** and **Q3**) to estimate an upper bound on the number of BGP updates related to RPKI, and showed that their impact—although sharply increasing in recent years—remains negligible compared to the overall volume of updates exchanged daily. Even during peaks, RPKI events account for less than 1% of total BGP updates. We also revealed that RPKI related updates are uniformly distributed in the global network, with no clear local imbalance among CPs. This work's encouraging findings, however, suggest multiple further investigations. Is it possible to identify types of RPKI events that leave a distinctive fingerprint on routing dynamics? Are there any CAs that can be considered primary contributors to RPKI induced updates? And finally, is the observed growth trend changing over time? We believe that the continued monitoring of the phenomena analyzed in this study is crucial for understanding the long-term implications of future extensive RPKI deployment. The source code employed in our measurements is publicly available at [26].

Acknowledgments

We warmly thank Randy Bush for providing the infrastructure and the resources used for the experiments of Sec. 3.3. This work was partially supported by the EU and Wallonia in the "Wallonie 2021-2027" program FEDER CyberGalaxia.

References

[1] Emile Aben. 2024. *RPKI Flutter.* https://observablehq.com/@emileaben/rpki-flutter-and-bgp

[2] Emile Aben. 2024. */rpki_flutter/daily/.* https://rd-www-1.ripe.net/rpki_flutter/daily/

[3] APNIC. 2026. *I-Rov Filtering Rate by Country.* https://stats.labs.apnic.net/rpki

[4] Jari Arkko, Michelle Cotton, and Leo Vegoda. 2010. IPv4 Address Blocks Reserved for Documentation. RFC 5737. doi:10.17487/RFC5737

[5] Hitesh Ballani, Paul Francis, and Xinyang Zhang. 2007. A study of prefix hijacking and interception in the Internet. *ACM SIGCOMM Computer Communication Review* 37, 4 (2007), 265–276.

[6] Tim Bruijnzeels, Oleg Muravskiy, Bryan Weber, and Rob Austein. 2017. The RPKI Repository Delta Protocol (RRDP). RFC 8182. doi:10.17487/RFC8182

[7] Randy Bush and Rob Austein. 2013. *The resource public key infrastructure (RPKI) to router protocol. RFC 6810 (Proposed Standard).* Technical Report. RFC Editor.

[8] Randy Bush, Keyur Patel, Dr. Philip F. Smith, and Mark Tinka. 2022. Policy Based on the Resource Public Key Infrastructure (RPKI) without Route Refresh. RFC 9324. doi:10.17487/RFC9324

[9] Wenqi Chen, Zhiliang Wang, Dongqi Han, Chenxin Duan, Xia Yin, Jiahai Yang, and Xingang Shi. 2022. ROV-MI: Large-Scale, Accurate and Efficient Measurement of ROV Deployment. In *Network and Distributed Systems Security (NDSS).*

[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. 2019. RPKI is Coming of Age: A Longitudinal Study of RPKI Deployment and Invalid Route Origins. In *Proceedings of the Internet Measurement Conference* (Amsterdam, Netherlands) *(IMC '19).* Association for Computing Machinery, New York, NY, USA, 406–419. doi:10.1145/3355369.3355596

[11] Federal Communications Commission. 2024. Reporting on Border Gateway Protocol Risk Mitigation Progress: Notice of Proposed Rulemaking. FCC NPRM, WC Docket No. 24-146. https://www.federalregister.gov/documents/2024/06/17/2024-13048/reporting-on-border-gateway-protocol-risk-mitigation-progress-secure-internet-routing Proposes periodic reporting by major U.S. broadband providers on BGP security measures, including RPKI/ROA deployment.

[12] Romain Fontugne, Amreesh Phokeer, Cristel Pelsser, Kevin Vermeulen, and Randy Bush. 2023. Rpki time-of-flight: Tracking delays in the management, control, and data planes. In *International Conference on Passive and Active Network Measurement.* Springer, 429–457.

[13] Yossi Gilad, Avichai Cohen, Amir Herzberg, Michael Schapira, and Haya Shulman. 2016. Are we there yet? On RPKI's deployment and security. *Cryptology ePrint Archive* (2016).

[14] Tomas Hlavacek, Haya Shulman, Niklas Vogel, and Michael Waidner. 2023. Keep Your Friends Close, but Your Routeservers Closer: Insights into RPKI Validation in the Internet. In *32nd USENIX Security Symposium (USENIX Security 23).* USENIX Association, Anaheim, CA, 4841–4858. https://www.usenix.org/conference/usenixsecurity23/presentation/hlavacek

[15] Geoff Huston. 2025. *A day in the life of BGP.* https://blog.apnic.net/2025/06/09/a-day-in-the-life-of-bgp/

[16] Geoff Huston. 2026. *potaroo.* https://bgp.potaroo.net/bgprpts/rva-index.html

[17] John Kristoff, Randy Bush, Chris Kanich, George Michaelson, Amreesh Phokeer, Thomas C. Schmidt, and Matthias Wählisch. 2020. On Measuring RPKI Relying Parties. In *Proceedings of the ACM Internet Measurement Conference* (Virtual Event, USA) *(IMC '20).* Association for Computing Machinery, New York, NY, USA, 484–491. doi:10.1145/3419394.3423622

[18] Weitong Li, Zhexiao Lin, Md. Ishtiaq Ashiq, Emile Aben, Romain Fontugne, Amreesh Phokeer, and Taejoong Chung. 2023. RoVista: Measuring and Analyzing the Route Origin Validation (ROV) in RPKI. In *Proceedings of the 2023 ACM on Internet Measurement Conference* (Montreal QC, Canada) *(IMC '23).* Association for Computing Machinery, New York, NY, USA, 73–88. doi:10.1145/3618257.3624806

[19] Ioana Livadariu, Romain Fontugne, Amreesh Phokeer, Massimo Candela, and Massimiliano Stucchi. 2024. A Tale of Two Synergies: Uncovering RPKI Practices for RTBH at IXPs. In *Passive and Active Measurement*, Philipp Richter, Vaibhav Bajpai, and Esteban Carisimo (Eds.). Springer Nature Switzerland, Cham, 88–103.

[20] MANRS. 2026. *MANRS ROAs Stats Tool.* https://observatory.manrs.org/#/roas/

[21] Pedro Marcos, Lars Prehn, Lucas Leal, Alberto Dainotti, Anja Feldmann, and Marinho Barcellos. 2020. AS-Path Prepending: there is no rose without a thorn. In *Proceedings of the ACM Internet Measurement Conference.* 506–520.

[22] Donika Mirdita, Haya Schulmann, Niklas Vogel, and Michael Waidner. 2023. The CURE To Vulnerabilities in RPKI Validation. In *Network and Distributed Systems Security (NDSS).*

[23] National Telecommunications and Information Administration. 2024. NTIA Supports FCC Internet Routing Security Proposal. NTIA Press Release. https://www.ntia.gov/press-release/2024/ntia-supports-fcc-internet-routing-security-proposal NTIA letter to the FCC endorsing the BGP routing security NPRM and urging broad RPKI/ROA adoption.

[24] "National Institute of Standards and Technology (NIST)". [n. d.]. *NIST RPKI Monitor, version 2.0.* https://rpki-monitor.antd.nist.gov/
[25] Office of the National Cyber Director, The White House. 2024. Roadmap to Enhancing Internet Routing Security. White House ONCD Report. https://bidenwhitehouse.archives.gov/wp-content/uploads/2024/09/Roadmap-to-Enhancing-Internet-Routing-Security.pdf Calls on U.S. network operators to adopt RPKI and publish ROAs covering their address space.
[26] Samuele Quinzi. 2026. *Codebase repository.* https://gitlab.com/uniroma3/compunet/networks/rpki-noise
[27] Y. Rekhter, T. Li, and S. Hares. 2006. *A Border Gateway Protocol 4 (BGP-4).* RFC 4271. RFC Editor.
[28] Andreas Reuter, Randy Bush, Italo Cunha, Ethan Katz-Bassett, Thomas C Schmidt, and Matthias Wählisch. 2018. Towards a rigorous methodology for measuring adoption of RPKI route validation and filtering. *ACM SIGCOMM Computer Communication Review* 48, 1 (2018), 19–27.
[29] RIPE. 2026. *Historic RPKI Archive.* https://rpki-study.github.io/rpki-archive/
[30] RIPE. 2026. Routing Information Service. https://ris.ripe.net/.
[31] Nils Rodday, Ítalo Cunha, Randy Bush, Ethan Katz-Bassett, Gabi Dreo Rodosek, Thomas C. Schmidt, and Matthias Wählisch. 2024. The Resource Public Key Infrastructure (RPKI): A Survey on Measurements and Future Prospects. *IEEE Transactions on Network and Service Management* 21, 2 (2024), 2353–2373. doi:10.1109/TNSM.2023.3327455
[32] Khwaja Zubair Sediqi, Romain Fontugne, Amreesh Phokeer, Massimiliano Stucchi, Massimo Candela, and Anja Feldmann. 2025. RPKI Syncing: Delay in Relying Party Synchronization. In *2025 9th Network Traffic Measurement and Analysis Conference (TMA).* IEEE, 1–11.
[33] Abtab Siddiqi. 2022. *KlaySwap – Another BGP Hijack Targeting Crypto Wallets.* https://manrs.org/2022/02/klayswap-another-bgp-hijack-targeting-crypto-wallets/
[34] Job Snijders. 2025. *RPKI's 2024 Year in Review.* https://labs.ripe.net/author/job_snijders/rpkis-2024-year-in-review/
[35] Job Snijders. 2026. *Welcome to RPKI Views!* https://www.rpkiviews.org/
[36] Job Snijders, Ben Maddison, Matt Lepinski, Derrick Kong, and Stephen Kent. 2024. A Profile for Route Origin Authorizations (ROAs). RFC 9582. doi:10.17487/RFC9582
[37] Massimiliano Stucchi. 2024. *A BGP Side Effect of RPKI.* https://labs.ripe.net/author/stucchimax/a-bgp-side-effect-of-rpki/
[38] Cecilia Testart, Philipp Richter, Alistair King, Alberto Dainotti, and David Clark. 2019. Profiling BGP Serial Hijackers: Capturing Persistent Misbehavior in the Global Routing Table. In *Proceedings of the Internet Measurement Conference (IMC '19).*
[39] Cecilia Testart, Philipp Richter, Alistair King, Alberto Dainotti, and David Clark. 2020. To Filter or not to Filter: Measuring the Benefits of Registering in the RPKI Today. In *International Conference on Passive and Active Network Measurement.* Springer, 71–87.
[40] University of Oregon. 1997. Route Views. http://www.routeviews.org/routeviews/.
[41] Koen van Hove, Jeroen van der Ham-de Vos, and Roland van Rijswijk-Deij. 2023. rpkiller: Threat Analysis of the BGP Resource Public Key Infrastructure. *Digital Threats* 4, 4, Article 58 (Oct. 2023), 24 pages. doi:10.1145/3617182
[42] Matthias Wählisch, Olaf Maennel, and Thomas C Schmidt. 2012. Towards detecting BGP route hijacking using the RPKI. *ACM SIGCOMM Computer Communication Review* 42, 4 (2012), 103–104.
[43] Matthias Wählisch, Robert Schmidt, Thomas C Schmidt, Olaf Maennel, Steve Uhlig, and Gareth Tyson. 2015. RiPKI: The tragic story of RPKI deployment in the Web ecosystem. In *Proceedings of the 14th ACM Workshop on Hot Topics in Networks.* 1–7.

Appendix

Ethics

This work only uses public data and does not raise any ethical consideration. The experiments described announcing prefixes in BGP have negligible impact on routing and were conducted for a short period of time.

Extra Material for Section 3.1

Family	Collector Peers Available By Year											
	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025
IPv4	18	19	16	17	41	99	127	94	77	73	78	68
IPv6	14	14	14	14	38	89	115	90	69	69	68	57

Table 1. CPs of RRC00 available for the analysis presented in this paper.

Extra Material for Section 3.3

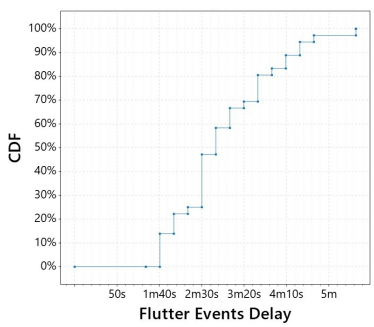


Fig. 14. CDF of the amount of delay between the exact timestamp of our beacon experiments and the corresponding flutter event.

Extra Material for Section 4.2.1

We prove that the snapshots referred to in Section 4.2 contain all the ROAs and the revocations, that the Archive can provide, that can affect the RPKI events happening during d. Consider again the example in Fig. 1. In fact, assume that the snapshots of days $d-1$, d, and $d+1$ were collected by Archive at times t_1, t_2, and t_3, respectively. Now, we show why the 3 snapshots mentioned above are *necessary* and *sufficient* to collect all the ROAs that may affect events during day d. Divide day d into two intervals, T_a that goes from the first midnight to t_2, and T_b that goes from t_2 to the second midnight. Let S_d be the snapshot of day d, and S_{d-1} and S_{d+1} the snapshots of days $d-1$ and $d+1$, respectively.

Necessity. S_{d-1} is *necessary* to capture ROAs that expire or are revoked during T_a (*e.g.*, R_5 and R_7). S_d is *necessary* to capture: (a) ROAs spanning the entire day d (*e.g.*, R_1); (b) ROAs that are issued during T_a and expire or are revoked during T_b (*e.g.*, R_9 and R_{16}); (c) ROAs that are issued during T_a and expire or are revoked before t_3 (*e.g.*, R_3); (d) ROAs that are issued after t_1 and expire or are

revoked during T_b (*e.g.*, R_2 and R_4). (e) Revocations happened during T_a that refer to ROAs that would have expired before t_3 (*e.g.*, ρ_5). S_{d+1} is *necessary* to capture: (a) ROAs that are issued during T_b (*e.g.*, R_6); (b) revocations that happened during T_b (*e.g.*, ρ_2).

Sufficiency. If a ROA that influenced events during d is present in any snapshot before S_{d-1}, then it is also present in snapshot S_{d-1}. In fact, consider a ROA R that influenced events during d and appears in any snapshot before S_{d-1}. To influence events in d, R must be of Type (1), (2), or (3), and therefore must be valid at the beginning of day d. Since S_{d-1} precedes d and R is present in a snapshot before (and so was issued before) S_{d-1}, then the validity period of R includes also the time of S_{d-1} and R is therefore present in snapshot S_{d-1}. Similarly, if a ROA or a revocation that influenced events during d is present in any snapshot after S_{d+1}, then it is also present in S_{d+1}. In fact, consider a ROA R that influenced events during d and appears in any snapshot after S_{d+1}. To influence events in d, R must be of Type (1) or (5), and therefore must be valid at the end of day d. Since S_{d+1} follows d and R is present in a snapshot after (and so expired or was revoked after) S_{d+1}, then the validity period of R includes also the time of S_{d+1} and R is therefore present in snapshot S_{d+1}. In case of revocations, a revocation that happened during d appears in all the snapshots subsequent to the revocation time that are included in the original validity period of the corresponding ROA. Since S_{d+1} follows d, if the revocation appears in a snapshot after S_{d+1}, then it means that the original validity period of the corresponding ROA includes S_{d+1}, hence the revocation appears also in S_{d+1}. In conclusion, the tree snapshots are sufficient to collect all the ROAs available in the Archive that influenced events during d.

Extra Material for Section 5.2

Fig. 15 shows the CDF of the estimated convergence times measured in our beacons experiments.

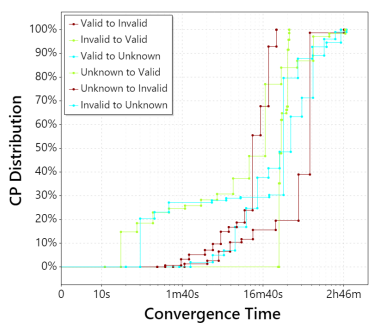
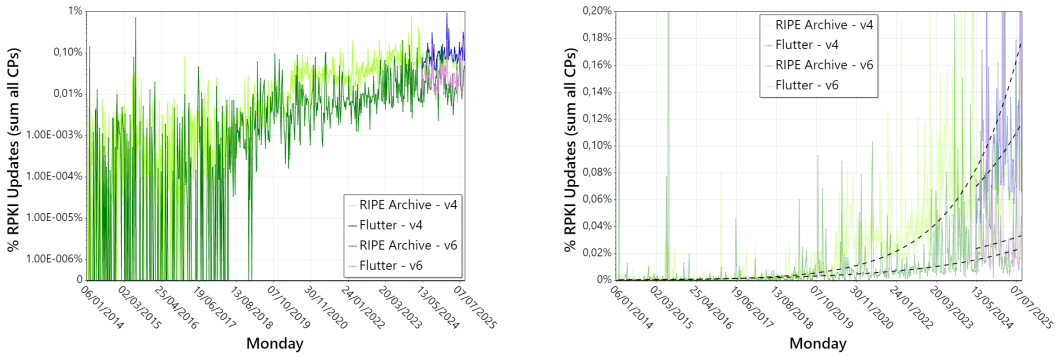


Fig. 15. CDF of BGP convergence times. The x-axis represents the convergence time (on a logarithmic scale). The y-axis shows the percentage of CPs that observed the last update before that time. Different curves describe the convergence time for different validation state transitions.

Extra Material for Section 6.1

Fig. 16a–b present the timeline of the percentage of RPKI-related updates over the total number of updates collected for all analyzed days. The plots display the entire period Jan 2014 – Aug 2025.

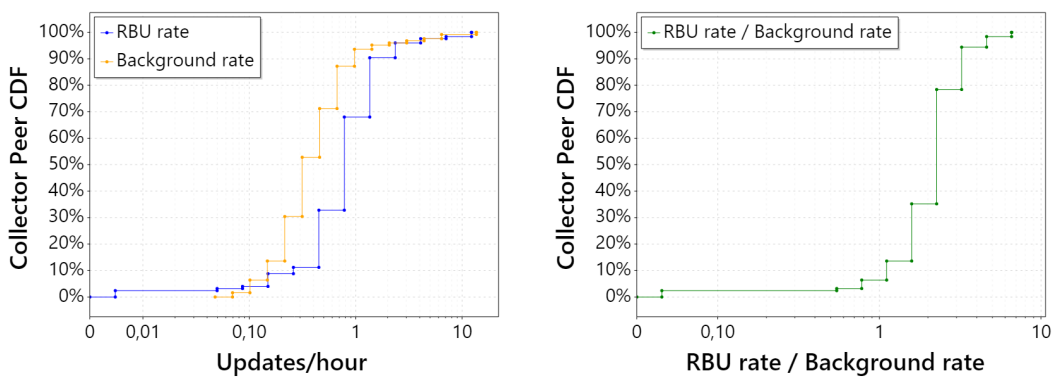


(a) Timeline showing the percentage of RPKI-related updates over the analyzed period (log-scale y).

(b) A detail of the percentage timeline in fig 16a with an exponential regression curve (linear scale y).

Fig. 16. Timelines showing the increase of the percentage of RPKI-related updates from Jan 2014 to Aug 2025.

Extra Material for Section 6.3



(a) CDFs of the update rates recorded by CPs. Blue: RPKI-related BGP Updates. Orange: Not RPKI-related BGP Updates. RIPE Archive Data.

(b) CDFs of the ratio between the BGP update rate during RPKI events and the BGP update rate outside RPKI events. RIPE Archive Data.

Fig. 17. Comparison between BGP update rates. RIPE Archive events.

Extra Material for Section 6.4 and Section 6.5

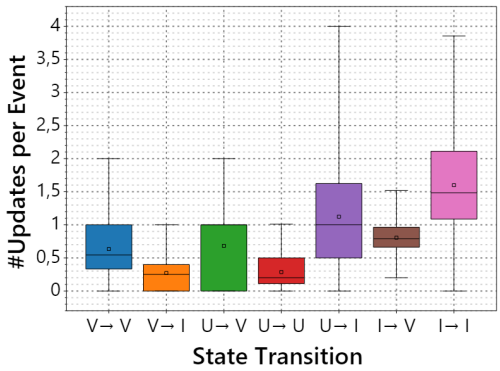


Fig. 18. Box-plot of the distribution over CPs of RPKI-related BGP updates per State Transition. Median, average, 25, and 75 percentiles are shown using boxes. A black square represents the average. States are: V=Valid, I=Invalid, and U=Unknown. An arrow between two states indicates the corresponding transition. RIPE Archive Data.

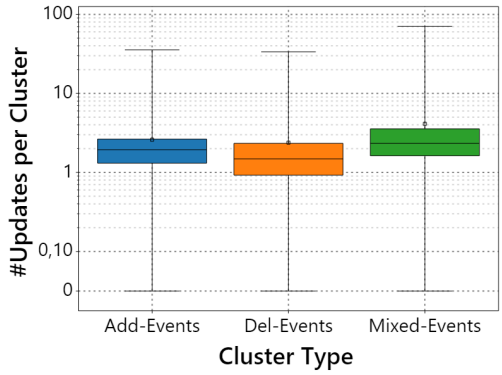


Fig. 19. Box-plot of the distribution over CPs of the number of RPKI-related BGP updates per cluster. Median, 25, and 75 percentiles are shown using boxes. A black square represents the average. The y-axis is log scale. Each box corresponds to a different type of cluster. RIPE Archive Data.

Received December 2025; accepted April 2026