University of Amsterdam

Research Project Two

Local Root Serving by Default: Quantifying the Traffic Trade-off Between Root Queries and Root Zone Distribution

January, 2026

Student:
Ilyas Rahimi
12330337

Supervisor:
dr. Marios Avgeris
Assistant Professor, University of Amsterdam
m.avgeris@uva.nl

CoSupervisor:
Willem Toorop
Senior Research Engineer, NLnet Labs
willem@nlnetlabs.nl

Abstract

Serving the DNS root zone locally at recursive resolvers has been proposed as a way to reduce operational dependence on the public root server system and to keep resolution working during outages. What is less clear in practice is how much extra traffic this creates when local-root-by-default is applied at scale. Moving from query-driven interaction with the root to an update-driven model replaces many small queries with fewer, larger transfers of zone data, and the net effect depends on how often the root zone changes and how resolvers obtain updates.

This study quantifies that trade-off using a measurement-based approach. On the root side, published RSSAC002 statistics are used to derive an estimate of conventional root query traffic in bytes per resolver per day. On the resolver side, packet captures from a controlled testbed are analysed for three widely deployed implementations (BIND, Unbound, and Knot Resolver), each operated with a local copy of the root zone using its native update mechanisms over DNS or HTTPS. For each configuration, the daily cost of keeping the local root in sync is expressed in the same unit and compared directly to the RSSAC-derived baseline.

The results show that, under current root zone change behaviour, local root operation consistently generates more traffic per resolver than conventional root resolution. The increase is dominated by update frequency and implementation choices, rather than by the transport protocol alone. A simple scaling exercise illustrates how these per-resolver differences accumulate when only a fraction of the global resolver population switches to local root, and confirms that the traffic impact remains non-negligible even under partial deployment.

Contents

1 Introduction

In order to replace the centrally managed HOSTS.TXT file and enable growth beyond a small research network, the Domain Name System (DNS) was introduced as a distributed and hierarchical naming mechanism [1, 2]. At the top of this hierarchy sits the DNS root. It acts as the point where resolution begins whenever a resolver cannot rely on locally cached data or intermediate delegations. Even though the root zone itself is limited in size and sees multiple updates per day, often without substantial content changes, the infrastructure that serves it operates under very different conditions. Recursive resolvers across the Internet still depend on it as part of normal resolution behavior, which means the root server system handles traffic at a scale that is far removed from the simplicity of the zone it publishes. This combination has made the root both a technical cornerstone of DNS operation and a recurring subject in discussions about robustness, dependency, and long-term resilience.

In recent years, serving the DNS root zone locally at recursive resolvers has gained renewed attention. Under this approach, resolvers maintain a local copy of the root zone and answer root-level queries themselves, rather than forwarding them to the public root server system. This practice is commonly referred to as local root serving. The operational motivations behind it are discussed in ongoing work within the IETF DNS Operations working group, most explicitly in the proposal to treat local root serving as a Best Current Practice [3]. The arguments in favor of this approach are mostly architectural. They focus on reducing reliance on external infrastructure, improving behavior during partial outages or routing disruptions, reducing certain security risks associated with redirected root query traffic, allowing operation in restricted or segmented environments, and giving operators more direct control over how resolution at the root level is handled [4].

What local root serving does not do is change the basic structure of DNS. The hierarchy remains the same, and correctness of the root zone remains critical. A resolver that serves the root locally is still dependent on receiving updates that are both timely and authentic. If updates are delayed for extended periods, the local copy may no longer reflect the authoritative state of the root zone. In practice, this can lead to stale delegation information and, in some cases, resolution failures. Rather than removing operational risk, local root serving mainly shifts the interaction with the root from continuous query exchange to periodic zone synchronization. Regular coordination with the root zone distribution mechanism therefore remains essential, even if some classes of failure are mitigated.

From an architectural perspective, the appeal of local root serving is easy to understand. Its impact on network traffic, however, is far less clear. In the conventional model, recursive resolvers generate root queries whenever cached delegation data expires or previously unseen names are resolved, contributing to the overall load on the root server system. In a local-root model, most of this query traffic disappears, but it is replaced by periodic transfers of root zone data. The resulting traffic balance depends on several factors, including how often the root zone changes, how large each update is, which update mechanism a resolver uses, and how resolvers behave under normal workloads. The net effect is not obvious *a priori*.

Although the resilience and policy aspects of local root serving have been discussed extensively, there is little empirical work that quantifies this traffic trade-off. Public measurements released under the RSSAC002 framework provide detailed insight into query volumes and traffic patterns observed at the root [5]. At the same time, modern resolver implementations such as BIND, Unbound, and Knot Resolver already support local root operation using different update mechanisms. What remains largely unexplored is how the traffic required to maintain a local copy of the root zone compares, in practice, to the volume of traffic generated by conventional root queries, and under which conditions one outweighs the other. Architecturally, local root serving is attractive, but its traffic implications have not yet been systematically measured.

This gap matters in part because of the role the root server system plays within the broader Internet governance structure. The Root Server System Advisory Committee (RSSAC) advises ICANN on issues related to root server operation, measurement, and capacity planning, and regularly publishes aggregated statistics describing observed root traffic behavior. These measurements feed directly into day-to-day operational choices, but they also shape broader architectural discussions about the future of the root server system. If local-root-by-default were to see wide adoption,

it would inevitably change the aggregate traffic patterns observed at the root. Understanding this effect is therefore relevant not only for resolver operators making deployment decisions, but also for how the root server system is observed, interpreted, and evolved over time.

What is missing in these discussions is not another architectural argument, but a clearer picture of how this shift behaves once it is put into operation. Local root serving replaces one form of interaction with another, but it does not eliminate interaction altogether. Query traffic gives way to update traffic, and assumptions are often made about how frequently updates occur or how costly they are in practice. How often the root zone actually changes, how much data is transferred per update, and how different resolver implementations handle this process are details that are usually treated as secondary, even though they directly determine the traffic outcome.

Seen from this angle, the central question is no longer whether local root serving is conceptually sound, but what it implies in concrete terms once deployed at scale. In particular, it becomes necessary to compare the traffic generated by periodic root zone distribution with the traffic produced by conventional, query-driven interaction with the root. Answering this requires looking at real update behavior over time, as well as at how resolver software implements and schedules root zone maintenance.

The Main Research Question: addressed in this work is therefore: *What is the traffic trade-off implied by local-root-by-default, comparing root query traffic in the conventional model to root zone distribution traffic required to maintain local copies at recursive resolvers?*

Subquestions:

1. *What is the empirical change behavior of the root zone over a long time horizon (e.g., changes per day and inter-change intervals), as observed via SOA serial evolution?*
2. *How many bytes are transferred per root zone change event under different update mechanisms (e.g., HTTPS file retrieval versus DNS-based transfer), and how variable is this cost over time?*
3. *How do major resolver implementations (BIND, Unbound, Knot Resolver) differ in their root zone maintenance behavior and update traffic cost, given the mechanisms they support?*
4. *Under realistic adoption scenarios, when does update traffic outweigh the reduction in root query traffic (and vice versa), and what is the implied break-even point?*

2 Related Work

Early work by Mockapetris established DNS as a distributed, hierarchical system in which delegation and caching enable global name resolution at Internet scale [1, 2]. Within this architecture, the root occupies a narrowly defined role. It serves as the starting point for delegation rather than as a repository of application data. As a result, much of the early literature treats the root as a structural necessity, not as an object of sustained operational analysis.

As the Internet expanded beyond its original research context, attention gradually shifted from the conceptual role of the root to its behavior as part of a globally deployed infrastructure. Measurement efforts and operational reporting began to focus on the traffic handled by the root server system and on how this traffic evolves over time. Today, this perspective is most clearly reflected in the work of the Root Server System Advisory Committee (RSSAC). RSSAC advises ICANN on matters related to root server operation, capacity planning, and measurement, and publishes regular reports describing observed traffic patterns at the root [5]. These publications present the root server system at an aggregate level. Rather than detailing the behavior of individual resolvers, they provide a combined view of how traffic at the root appears when considered as a whole, and are commonly used as reference points in discussions of root load and operational capacity.

At the same time, the scope of these measurements is intentionally constrained. RSSAC reports describe what is observed at the root servers under the current resolution model, but they do not attempt to explore alternative architectures or to predict how traffic patterns might shift if resolver behavior were to change. In particular, they do not quantify the traffic associated with maintaining local copies of the root zone at recursive resolvers, nor do they offer a per-resolver

view that compares query-driven traffic with update-driven traffic. As a result, RSSAC data is highly valuable for understanding the present state of the system, but insufficient on its own to evaluate the traffic implications of architectural changes.

One such architectural change is local root serving. The idea of serving the DNS root zone locally at recursive resolvers has existed for some time, but has gained renewed attention in recent standardization work within the IETF DNS Operations working group. The proposal to recognize local root serving as a Best Current Practice presents it as an incremental adjustment to existing DNS operation rather than as a redesign of the protocol [3]. The draft focuses on operational considerations, including reduced reliance on external infrastructure, improved behavior during routing disruptions or denial-of-service events, and continued operation in constrained or partially disconnected environments. Notably, it does not frame local root serving as a performance optimization, nor does it claim that it reduces overall traffic. The motivation remains architectural and operational in nature.

At the same time, the proposal makes clear that local root serving does not remove the root from the DNS hierarchy or alter the delegation model on which DNS is based. Correctness of the root zone remains essential, and resolvers that serve the root locally remain dependent on receiving updates that are both timely and authentic. In effect, the proposal replaces continuous query exchange with periodic distribution of zone data as the primary interface with the root. While this shift is motivated by robustness and operational control, it also introduces a different dependency whose traffic implications are not examined in detail in the standardization documents.

From an implementation perspective, local root serving is no longer hypothetical. Widely deployed recursive resolver implementations, including BIND, Unbound, and Knot Resolver, already support operation with a local copy of the root zone. These implementations differ in how updates are obtained and applied, reflecting distinct design decisions and assumptions about operational practice. Documentation and operational guidance describe how local root support can be enabled and maintained, but they do not quantify the resulting traffic cost or compare it to the query traffic generated under the conventional resolution model.

A similar pattern can be observed elsewhere in the DNS ecosystem. Resolver behavior implemented in software often precedes formal standardization, with documents emerging to describe and align existing practice rather than to propose entirely new designs. Local root serving fits this pattern. It is being discussed as a best practice after support has already been implemented in major resolver software, not as a speculative idea. What remains missing is a systematic analysis of what this practice implies for network traffic when considered beyond individual deployments.

Existing work establishes the architectural motivation for local root serving, documents the operational behavior of the root server system under the conventional model, and shows that resolver software already supports alternative modes of operation. What it does not provide is a quantitative connection between these pieces. In particular, there is little empirical work that compares the traffic generated by conventional, query-driven interaction with the root to the traffic required to distribute and maintain local root zone copies over time. This unresolved question motivates a measurement-driven examination of how query traffic and update traffic relate in practice, and forms the basis for the methodology described in the following section.

3 Methodology

The objective of this study is to express the traffic implications of local root operation in a form that allows direct comparison with the conventional, query-driven use of the public root server system. Traffic is therefore treated consistently as bytes transferred per resolver per day. The methodology combines two complementary perspectives: aggregate observations of traffic at the root server system, and direct measurements of root zone distribution traffic at recursive resolvers.

For the conventional resolution model, measurement data published under the RSSAC002 framework is used[6]. These measurements are produced by the root server operators and aggregated by the Root Server System Advisory Committee. They describe traffic as observed at the root servers and include both query counts and packet size distributions across multiple transport protocols. Data is reported per root server letter and per day, and distinguishes between UDP, TCP, and DNS over TLS traffic, as well as between requests and responses.

To estimate traffic volume, two classes of RSSAC002 metrics are combined. Traffic-volume measurements provide the total number of DNS queries and responses handled by each root server over a given period, separated by protocol and address family. Traffic-sizes measurements provide distributions of packet sizes, reported in buckets. By combining query counts with the corresponding size distributions, an estimate of the total number of bytes transferred by each root server during a given day can be derived. For each size bucket, the upper bound is used as a conservative approximation of packet size. This approach includes both DNS payload and protocol overhead, reflecting traffic as it appears on the wire.

Traffic volume is computed separately for each root server letter. To obtain a per-resolver estimate, this volume is related to the number of resolvers interacting with each root. RSSAC002 reports unique sources as distinct IPv4 addresses and aggregated IPv6 sources. Dividing the total daily traffic volume per root by the corresponding number of unique resolver sources yields an average number of bytes served per resolver per day for that root. This calculation preserves differences between root servers rather than collapsing the entire system into a single average.

The analysis is applied to a seven-day window, covering January 1 through January 7. When the values are compared day by day within this interval, no strong fluctuations are observed. Differences between days are small and follow the same overall pattern. For this reason, the average over the seven-day period is used as a practical representation of typical root query traffic per resolver.

In contrast to the root-side analysis, traffic on the resolver side is observed directly. The evaluation focuses on three recursive resolver implementations that are widely deployed in operational networks: BIND, Unbound, and Knot Resolver. All three already provide support for maintaining a local copy of the root zone. The focus on open-source software makes it possible to reason about configuration choices and update behavior without relying on undocumented assumptions.

Each resolver is operated with a local copy of the root zone, using the update mechanisms that are natively supported by the implementation. Depending on the resolver, this results either in DNS-based transfer of zone data or in retrieval over HTTPS. During root zone update events, packet captures are collected and analyzed. All traffic associated with the update process is included, including transport-layer overhead, so that the measured volumes reflect actual bytes transferred.

Packet captures were collected over a four-day period. During this time, root zone updates follow a regular pattern. Both the timing of updates and the amount of data transferred show little variation from one day to the next. While the measurement window is limited, the stability of the root zone and its update process makes it unlikely that a longer observation period would lead to qualitatively different results.

For each resolver and update mechanism, the total number of bytes transferred per update is calculated, along with the effective daily traffic cost of maintaining a local root copy. These values are then compared directly with the per-resolver traffic estimates derived from RSSAC002 data. Both sides of the comparison are expressed in bytes per resolver per day, allowing a direct assessment of the traffic trade-off between query-driven and update-driven interaction with the DNS root.

Several aspects are explicitly outside the scope of this methodology. The computational cost of DNSSEC validation, resolver caching behavior beyond root zone maintenance are not considered. The focus is strictly on traffic volume. Combined, these steps make it possible to compare conventional root query traffic with the traffic required to maintain a local root copy using the same unit of measurement. Both perspectives are derived from observed behavior rather than abstract models, which allows the traffic trade-off to be examined without relying on assumptions that cannot be verified.

4 Experimental Setup

4.1 Experimental Environment

The local-root side of the traffic trade-off was evaluated in a controlled virtual testbed. All experiments were run on a Proxmox host that served purely as a virtualization platform. No assumptions are made about the characteristics of the physical uplink, since the focus of this work is on the

volume of data exchanged to maintain local copies of the root zone, not on latency, throughput, or end-to-end performance. Within this environment, all components relevant to the experiment were isolated in separate virtual machines and communicated exclusively over a local virtual network.

4.2 Resolver Instances and Roles

Four virtual machines were deployed, each with a single, well-defined role. One VM ran BIND configured as a recursive resolver operating with a local copy of the root zone. A second VM ran Unbound in a comparable configuration, using DNS-based retrieval of the root zone. A third VM ran a separate Unbound instance that obtained the root zone over HTTPS rather than via DNS transfer, following the mechanisms described in the local-root best current practice draft. The fourth VM ran Knot Resolver, again configured to serve the root locally.

All systems ran a recent Linux distribution, and only open-source resolver software was used throughout the experiment. This choice avoids reliance on vendor-specific behavior that cannot be inspected and ensures that configuration and update behavior can be described precisely and reproduced.

4.3 Local Root Configuration and Update Mechanisms

Each resolver was configured according to the guidance in the local-root best current practice draft and the official documentation of the respective resolver software. For BIND and for the Unbound instance that relies on DNS, the resolver operates in a secondary-style role with respect to the root zone and interacts with an authoritative source using standard DNS mechanisms. In both cases, the resolver checks the SOA record of the zone to determine whether an update may be available and retrieves zone data using IXFR or AXFR, depending on what is supported by the authoritative side.

Knot Resolver was configured using its built-in local root functionality, which likewise treats the resolver as a consumer of zone data rather than as a pure client issuing iterative queries to the public root server system. The HTTPS-based Unbound instance obtained the same root zone via HTTPS from a configured web endpoint instead of using DNS zone transfer. Conceptually, this instance fulfills the same role as the DNS-based configurations, but it interacts with a different distribution mechanism.

4.4 Update Scheduling and SOA Semantics

The update behavior of the resolvers is driven by the timers published in the SOA record of the production root zone. In the root zone, the refresh field is set to 1800 seconds, corresponding to a thirty-minute interval. A resolver that serves a local copy of the root and behaves like a secondary is expected to use this timer to decide when to check for a newer version of the zone.

In the experimental setup, the DNS-based BIND and Unbound instances follow this model. At the end of each refresh interval, the resolver checks the SOA record. If the serial number has not changed, no transfer takes place. When the serial has advanced, only the changed portion of the zone is retrieved using IXFR where possible.

The HTTPS-based Unbound instance applies the same refresh interval, but cannot issue SOA queries to a web server. As a result, it downloads the full root zone over HTTPS every time the refresh interval expires. Knot Resolver follows its own internal scheduling logic in the configuration used here and checks for updates much less frequently, resulting in approximately one full update per day.

4.5 Traffic Measurement Method

To measure the traffic cost associated with these behaviors, packet captures were recorded on the resolver virtual machines themselves. For each resolver, capturing was enabled on the primary network interface that carried DNS and HTTPS traffic related to root zone maintenance. No capture-time filters were applied, so that protocol overhead such as TCP connection setup, TLS handshakes, and SOA queries are included in the accounting.

During analysis, only traffic associated with root zone updates and the control queries that trigger them was selected, but the underlying capture files contain the complete traffic stream as observed by the resolver. This ensures that the reported byte counts reflect the real cost of maintaining a local root copy, rather than an idealized view that excludes transport-layer overhead.

4.6 Observation Window and Stability

Each resolver configuration was observed over a period of four consecutive days. Throughout this window, the resolvers followed their configured update behavior without manual interaction or induced changes. Root zone updates were driven by the actual serial evolution of the production root zone, rather than by synthetic or forced changes.

Across the observation window, the packet traces show stable and repeatable patterns that align with the expected behavior of each resolver. BIND and DNS-based Unbound check for updates at intervals derived from the SOA refresh value and only transfer data when the serial changes. Knot Resolver performs a single update in a roughly consistent daily time window. The HTTPS-based Unbound instance attempts a full zone download every thirty minutes, resulting in substantially larger capture files. Although absolute byte counts differ between resolver implementations, the structure of the traffic remains consistent across days, which supports the use of average daily traffic per resolver as a representative value.

4.7 Relation to the Comparative Model

This experimental setup provides a controlled view of how widely deployed recursive resolver implementations behave when configured to serve the DNS root locally, and how much traffic they generate to keep their local copy synchronized. All measurements include transport and protocol overhead and are grounded in the actual timing behavior dictated by the root zone's SOA parameters. This makes it possible to compare the traffic cost of local-root operation observed in the testbed directly with per-resolver traffic estimates derived from measurements of the conventional, query-driven root server model.

5 Results and Analysis

This section presents the measured traffic volumes associated with local root zone operation at recursive resolvers and compares them with traffic observed in the conventional, query-driven root server system. All values are expressed in megabytes per resolver per day, in line with the methodology described in Section 3. The observation window for local-root measurements spans four days, from January 20 through January 23. Conventional root traffic is analyzed over a seven-day window, from January 1 through January 7.

5.1 Conventional Root Traffic (RSSAC002)

To provide a baseline for comparison, traffic associated with conventional query-driven interaction with the public root server system was analyzed using RSSAC002 data. The analysis covers January 1 through January 7. Traffic volume and resolver counts are combined to derive per-resolver traffic expressed in megabytes per day.

Table 1: Conventional root traffic per resolver (RSSAC002), MB/day

Day	b	c	d	f	h	i	j	k	l	m
1	0.69	0.71	1.04	11.40	0.88	1.30	1.11	1.06	1.35	0.70
2	0.69	0.72	1.08	11.26	0.90	1.27	1.08	1.04	1.35	0.68
3	0.66	0.70	1.03	10.92	0.86	1.25	1.07	1.02	1.32	0.62
4	0.66	0.70	1.04	10.91	0.85	1.30	1.05	1.07	1.34	0.69
5	0.66	0.70	1.06	11.13	0.88	1.25	1.01	1.07	1.31	0.63
6	0.68	0.71	1.06	11.35	0.89	1.22	0.97	1.06	1.33	0.67
7	0.68	0.72	1.06	11.28	0.90	1.25	0.99	1.10	1.36	0.70
Average	**0.67**	**0.71**	**1.05**	**11.18**	**0.88**	**1.26**	**1.04**	**1.06**	**1.34**	**0.67**

5.2 Local Root Traffic

5.2.1 BIND (DNS-based IXFR)

For BIND, root zone updates are triggered exclusively by SOA serial changes. Each incremental update observed during the measurement period transfers approximately 1.45 MB. Table 2 summarizes the number of observed updates per day and the resulting daily traffic volume.

Table 2: BIND local root traffic

Date (2026)	Updates	MB per update	Total MB/day
Jan 20	2	1.45	2.90
Jan 21	3	1.45	4.35
Jan 22	4	1.45	5.80
Jan 23	3	1.45	4.35
Average	–	–	**4.35**

Across the four-day window, the average traffic required by BIND to maintain a local copy of the root zone is 4.35 MB per resolver per day.

5.2.2 Unbound (DNS-based IXFR)

Unbound configured with DNS-based transfer follows the same update logic as BIND, but transfers a slightly smaller amount of data per update. Each observed incremental update transfers approximately 1.31 MB. The resulting daily traffic volumes are shown in Table 3.

The average daily traffic for DNS-based Unbound over the observation window is 3.93 MB per resolver per day.

Table 3: Unbound (DNS-based) local root traffic

Date (2026)	Updates	MB per update	Total MB/day
Jan 20	2	1.31	2.62
Jan 21	3	1.31	3.93
Jan 22	4	1.31	5.24
Jan 23	3	1.31	3.93
Average	–	–	**3.93**

5.2.3 Unbound (HTTPS-based)

When Unbound retrieves the root zone over HTTPS, it cannot rely on SOA semantics to detect changes. As a result, it downloads the full root zone at every refresh interval. In the configuration used here, this occurs every thirty minutes, resulting in 48 downloads per day, each transferring approximately 2.19 MB.

Table 4: Unbound (HTTPS-based) local root traffic

Date (2026)	Updates/day	MB per update	Total MB/day
Jan 20	48	2.19	105.12
Jan 21	48	2.19	105.12
Jan 22	48	2.19	105.12
Jan 23	48	2.19	105.12
Average	–	–	**105.12**

The daily traffic volume is dominated by the fixed polling interval rather than by actual root zone changes.

5.2.4 Knot Resolver (HTTPS-based)

Knot Resolver also retrieves the root zone via HTTPS, but applies a much less frequent update schedule. In the configuration tested here, Knot performs approximately one full update per day.

Table 5: Knot Resolver local root traffic

Date (2026)	Time (CET)	Data transferred (MB)
Jan 20	12:06	2.69
Jan 21	12:06	2.61
Jan 22	12:06	2.67
Jan 23	12:06	2.65
Average	–	**2.65**

Despite using full zone transfers, Knot Resolver generates the lowest average daily traffic among the evaluated resolvers due to its low update frequency.

5.3 Scaling impact of partial deployment of local root resolvers

To relate the per-resolver measurements to system-wide effects, the observed differences were scaled to a hypothetical partial deployment scenario. Rather than assuming a full migration, three adoption levels were considered: 10%, 20%, and 30% of all resolvers switching from conventional root querying to a local-root configuration.

The baseline for comparison is the average traffic volume per resolver per day observed in the RSSAC002 data, which is approximately 2 MB, the average of all roots. For each resolver

implementation, the difference between its measured local-root traffic and this baseline was taken as the per-resolver delta. The resolver population was estimated at 59,137,348 based on the sum of unique sources reported across all root servers in the same RSSAC002 dataset, treating each source as an approximate resolver instance.

Figure 1 shows the resulting additional traffic volume per day, expressed in terabytes, for each resolver implementation and adoption level. The vertical axis is logarithmic to account for the large differences in magnitude between the implementations.

Resolvers using DNS-based zone transfer mechanisms (BIND and DNS-based Unbound) show moderate increases in traffic as adoption rises. Knot Resolver, which retrieves the root zone via HTTPS but performs updates only once per day, results in the smallest increase across all scenarios. In contrast, HTTPS-based Unbound, even when assuming corrected behavior where updates are triggered by actual root zone changes rather than fixed polling, produces a substantially larger increase in aggregate traffic due to the size of full zone transfers.

The scaling behavior is linear with respect to the adoption fraction, as expected, but the absolute magnitude differs significantly between resolver implementations. These results quantify how implementation choices translate into system-wide traffic effects when local root operation is deployed beyond isolated instances.

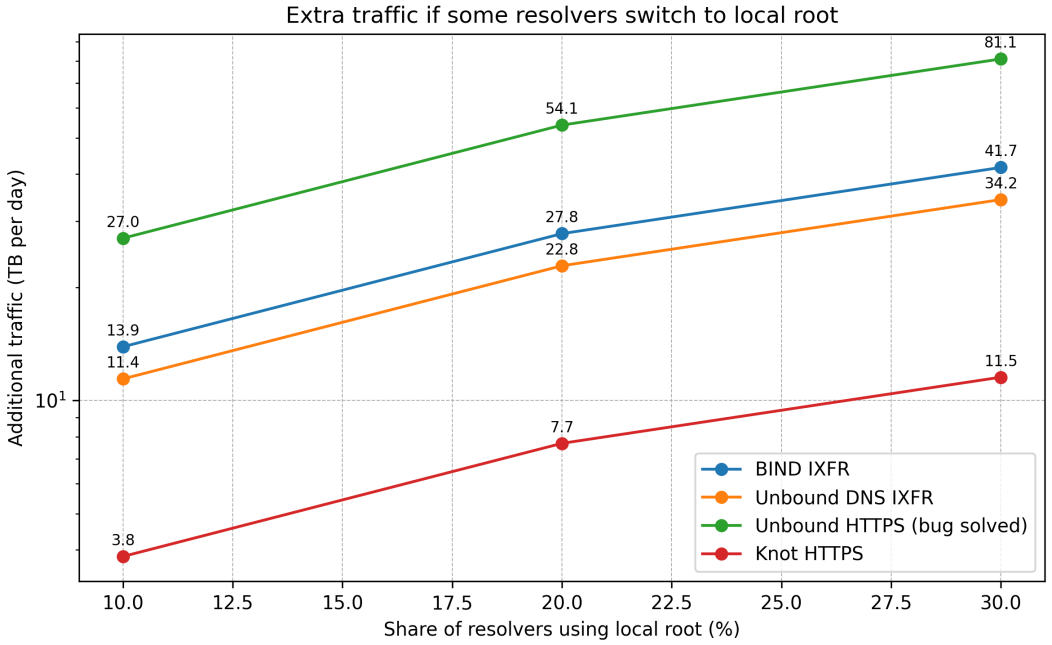

Figure 1: Additional daily traffic under partial deployment of local root resolvers, shown relative to the conventional root query baseline. The values represent total traffic across the entire resolver population in the scenario, rather than per-root-server traffic.

6 Discussion: Local Root Traffic in Context

The results presented earlier make it possible to place the traffic impact of local root operation next to the conventional, query-driven interaction with the public root server system in a concrete and measurable way. Because both perspectives are expressed in bytes per resolver per day, the comparison does not rely on abstract ratios or trends, but on quantities that can be compared directly.

What becomes visible from the measurements is a shift in how traffic is generated. In the conventional model, resolvers interact with the root through a continuous stream of small queries. Each individual exchange is lightweight, but together they form a steady background load. With

local root operation, this pattern changes. Most query traffic disappears and is replaced by occasional transfers of root zone data. These transfers are larger in size, but occur far less frequently. Whether this leads to more or less traffic overall depends primarily on how often the root zone changes and on how resolvers react to those changes.

6.1 Local Root Traffic per Resolver

Over the four-day observation period, the DNS-based local root configurations show a stable and predictable pattern. Both BIND and Unbound, when using DNS-based zone transfer, retrieve data only when the root zone serial advances. The amount of data transferred during each update remains nearly constant, and day-to-day variation in traffic can be traced directly to differences in how many root zone changes occur on a given day.

For BIND, each update involves a transfer of roughly 1.45 MB. Days with few changes therefore result in modest traffic volumes, while days with more frequent changes produce proportionally higher totals. When these daily values are averaged across the observation window, the resulting traffic cost amounts to several megabytes per resolver per day. DNS-based Unbound follows the same mechanism, but with a slightly smaller transfer size of approximately 1.31 MB per update, leading to a lower daily total under otherwise identical conditions.

What stands out here is that the resolver implementation itself plays only a secondary role. Once the transfer size per update is fixed, the long-term traffic cost is shaped mainly by how often updates occur. In practice, the cadence of root zone changes has a stronger influence on traffic volume than differences between BIND and Unbound at the protocol level.

6.2 HTTPS-based Retrieval and Update Logic

The HTTPS-based Unbound configuration behaves differently from the DNS-based setups. In the configuration observed here, Unbound retrieves the full root zone over HTTPS at every refresh interval, without first determining whether the zone has changed. As a result, traffic is generated on a fixed schedule, independent of actual root zone dynamics.

Each transfer is approximately 2.19 MB. When repeated at thirty-minute intervals, this leads to a daily traffic volume that is dominated by the polling behavior itself rather than by root zone activity. Compared to the DNS-based configurations, this produces a substantially higher number of bytes transferred per resolver per day.

This behavior should not be interpreted as an inherent property of HTTPS as a transport mechanism. Instead, it follows from the update logic currently in use. As confirmed in a discussion with senior Unbound developer (personal communication), this behavior is considered a bug and is expected to be corrected. The intended design is for HTTPS-based retrieval to take the root zone serial into account, so that full downloads only occur when an actual change is detected. Once corrected, the traffic profile of HTTPS-based Unbound would be expected to move closer to that of the DNS-based implementations, with remaining differences mainly reflecting transport overhead rather than update frequency.

6.3 Knot Resolver and Conservative Update Strategy

Knot Resolver also retrieves the root zone over HTTPS, but its observed behavior differs clearly from that of the HTTPS-based Unbound configuration. In the setup evaluated here, Knot performs roughly one full root zone update per day, resulting in a daily traffic volume of approximately 2.6 to 2.7 MB per resolver.

This behavior reflects an explicit design choice. According to the Knot Resolver documentation[7], the developers deliberately avoid a literal interpretation of RFC 7706 that would involve frequent polling for root zone updates. Instead, Knot adopts a more restrained update strategy that limits how often the root zone is retrieved, even when changes may occur more frequently.

The internal considerations that lead to the once-per-day update schedule are not fully exposed by the measurements alone. Factors such as caching behavior or time-to-live handling may influence the observed pattern, but these details cannot be disentangled with certainty from the available data. What can be stated with confidence is that Knot represents a distinct design approach. The

results show that maintaining a local copy of the root zone does not inherently require frequent updates or large recurring transfers; update frequency is ultimately a matter of design choice rather than technical necessity.

6.4 Comparison with Conventional Root Traffic

The RSSAC002-based analysis provides a complementary view of resolver behavior under the conventional, query-driven model. When the reported measurements are converted into bytes per resolver per day, the resulting values remain around one or two megabytes, with the only outlier being f-root.

When placed alongside the local root measurements, the contrast becomes clear. All local root configurations studied here generate more traffic per resolver than the conventional model, although the magnitude of the difference varies significantly. DNS-based local root operation results in several megabytes per resolver per day, while more aggressive polling strategies amplify this effect further.

This difference should not be read as a judgement about efficiency or design quality. Rather, it reflects a structural difference in how traffic is exchanged. Conventional resolution spreads traffic across many small interactions, while local root operation concentrates traffic into fewer, larger transfers. From a traffic volume perspective alone, the latter is more costly per resolver under the current rate of root zone change.

6.5 Scaling Effects and Partial Deployment

The per-resolver differences become more consequential when considered at scale. Using published estimates of the number of resolvers interacting with the root server system, it is possible to explore how aggregate traffic would change if only a fraction of these resolvers were to switch to local root operation.

Even partial deployment scenarios lead to noticeable increases in total traffic associated with root zone distribution. Because the additional traffic per resolver is positive for all local root configurations relative to the conventional baseline, the overall effect scales linearly with adoption. The choice of update strategy plays a decisive role. Conservative approaches, such as the one observed for Knot Resolver, result in comparatively small increases, while frequent polling can lead to much larger aggregate effects.

At the same time, the measurements indicate that the amount of data transferred per update remains largely constant. This suggests that long-term changes in aggregate traffic are driven mainly by update frequency rather than by variation in zone size or resolver-specific behavior.

6.6 Limitations

This discussion is based on a limited observation window for local root measurements. While the stability of the root zone suggests that the qualitative conclusions are unlikely to change over longer periods, additional measurements could refine the estimated averages.

Furthermore, the analysis is intentionally limited to traffic volume. Other aspects, such as latency, operational complexity, or mixed deployment scenarios, are outside the scope of this work. The results should therefore be interpreted as an assessment of traffic implications only, rather than as a comprehensive evaluation of local root operation.

6.7 Threats to Validity

Any measurement-based study is shaped by the choices made in data selection, observation period, and experimental setup. The results presented here should therefore be interpreted with these constraints in mind.

On the side of conventional root traffic, the analysis relies on RSSAC002 data, which is published periodically and reflects aggregated observations at the root server system. Because this study was time-bounded, the analysis focuses on a seven-day window in early January. Within this window, the reported values are stable and follow a consistent pattern from day to day. While this suggests

that the selected period is representative, a longer observation period could still provide additional reassurance that no atypical behavior was missed.

For the local root measurements, traffic was observed over a four day period in a controlled testbed. During this time, the amount of data transferred per update was consistent across days for all resolver implementations. This indicates that the measured values are not the result of short-lived fluctuations. Nevertheless, observing the same setups over a longer time span could further confirm that this behavior persists under less common update scenarios.

This setup makes it possible to account for all bytes exchanged during updates, including protocol and transport overhead. At the same time, the environment remains simplified when compared to operational networks, where traffic conditions and configurations can be more diverse.

7 Conclusion

This study examined the traffic implications of operating recursive resolvers with a local copy of the DNS root zone, compared to the conventional model in which resolvers interact with the public root server system through queries. By expressing both models in the same unit bytes per resolver per day the comparison can be made directly, without relying on indirect indicators or relative trends.

Across all configurations that were measured, local root operation consistently results in a higher amount of data transferred per resolver than conventional root resolution. This outcome is not tied to any single resolver implementation, but follows from the structural change introduced by local root operation. In the conventional model, resolvers generate traffic through many small exchanges spread over time. When a local root is used, most of this query traffic disappears and is replaced by fewer, larger transfers associated with maintaining the root zone itself.

The measurements further show that the size of these transfers is remarkably stable. For DNS-based zone transfers, both BIND and Unbound retrieve a similar amount of data whenever the root zone changes. As a result, differences in daily traffic volume are primarily explained by how often the root zone serial changes, rather than by variations in resolver behavior or protocol efficiency. Once the update size is fixed, update frequency becomes the dominant factor shaping long-term traffic cost.

The HTTPS-based Unbound configuration observed in this study behaves differently. In the measured setup, Unbound retrieves the full root zone at fixed intervals without first determining whether a change has occurred. This leads to traffic patterns that are driven by polling frequency rather than by actual root zone evolution. As confirmed in discussion with a senior Unbound developer (personal communication), this behavior is considered a bug rather than an intended design choice. Once corrected, HTTPS-based retrieval is expected to follow the same change-driven logic as DNS-based transfers, bringing its traffic profile much closer to the other implementations.

Knot Resolver illustrates that local root operation does not inherently require frequent updates. In the configuration evaluated here, Knot retrieves the root zone roughly once per day. This keeps daily traffic volumes relatively low, but also means that changes to the root zone are not necessarily reflected immediately at the resolver. This behavior aligns with the design philosophy described in the Knot Resolver documentation, where RFC 7706 is not treated as a requirement to continuously poll for updates [7]. Instead, the resolver adopts a restrained update schedule, accepting reduced freshness in exchange for lower update traffic.

Beyond traffic volume, local root operation also changes the qualitative properties of resolver behavior. By eliminating routine queries to external root servers, resolvers reduce the exposure of query names on the network and limit the amount of resolver activity visible to upstream infrastructure. In addition, maintaining the root locally reduces dependence on the availability of remote root servers, mitigating certain operational risks such as network disruption or denial-of-service attacks against the public root infrastructure [8, 9]. These benefits are tangible, but they coexist with the increased data transfers required to keep a local copy of the root zone up to date.

When the per-resolver effects are considered at scale, even partial deployment leads to noticeable changes. Because the additional traffic per resolver is positive across all local root configurations measured, aggregate traffic associated with root zone distribution increases linearly with adoption. At the same time, the measurements indicate that the amount of data transferred per update

remains largely constant. This suggests that system-wide impact is shaped primarily by update frequency, rather than by gradual growth of the zone or incremental protocol optimizations.

Several limitations frame these conclusions. The local root measurements cover a relatively short observation period, and longer-term data could refine the averages reported here. The analysis is also limited to traffic volume. Other considerations, such as latency, operational complexity, deployment diversity, or policy implications, are outside the scope of this work. The results should therefore be read as a focused assessment of traffic implications, not as a definitive judgement on whether local root operation should or should not be deployed.

Research Questions Revisited

The main research question addressed in this work is:

What is the traffic trade-off implied by local-root-by-default, comparing conventional root query traffic with the root zone distribution traffic required to maintain local copies at recursive resolvers?

This question is answered by combining two complementary measurement perspectives. Traffic observed at the public root server system is analyzed using RSSAC002 data and expressed as bytes per resolver per day. Direct measurements at recursive resolvers quantify the traffic generated by maintaining a local copy of the root zone under different update mechanisms. Expressing both sides in the same unit makes it possible to compare query-driven and update-driven interaction with the root on equal terms. The results show that local-root-by-default shifts traffic from many small queries to fewer, larger transfers, leading to a higher traffic volume per resolver under current root zone change rates.

The first subquestion asks how the root zone changes over time. This is addressed by observing the evolution of SOA serial numbers, which shows that the root zone changes multiple times per day, but not at fixed intervals. The size of the zone remains stable, while update frequency varies from day to day, establishing update timing as the primary driver of local root traffic.

The second subquestion concerns the amount of data transferred per update. Packet-level measurements show that transfer size per update is stable for each mechanism, both for DNS-based transfers and for HTTPS-based retrieval. Variability in daily traffic is therefore explained by how often updates occur, not by fluctuations in transfer size.

The third subquestion compares resolver implementations. Measurements of BIND, Unbound, and Knot Resolver show that differences in update logic and design philosophy have a stronger influence on traffic outcomes than transport choice alone. Implementations that closely follow serial changes generate more frequent traffic, while conservative update strategies reduce traffic at the cost of delayed freshness.

The final subquestion addresses deployment at scale. By combining per-resolver traffic differences with published estimates of resolver population size, the analysis shows that aggregate update traffic increases linearly with adoption. Under current conditions, even moderate adoption levels result in higher total traffic than the conventional model, indicating that the break-even point is not reached with present root zone change frequencies and update strategies.

These results show that the traffic trade-off of local-root-by-default is predictable and measurable. It is shaped primarily by update frequency and implementation choices, rather than by transport mechanisms alone. Local root operation offers clear qualitative advantages, but these come with a traffic cost that must be weighed in context.

8 Future Work: Incremental Signing and Root Zone Distribution

One factor that strongly shapes the traffic cost of local root operation is the way in which the root zone is currently signed. Because the root zone is produced and signed as a single, internally consistent unit, resolvers that retrieve the root over non-DNS mechanisms effectively need to download a complete, self-consistent version of the zone whenever an update is performed. Even when only a small portion of the zone changes, the cryptographic structure ties the data together in a way that prevents straightforward incremental distribution.

In principle, this limitation follows from how the root zone is produced and signed today, rather than from local root operation itself. If the root zone were signed in a more incremental manner, it could become possible to distribute updates in smaller, independently verifiable parts. As a conceptual illustration, if signing and publication were structured such that changes could be applied in segments spread over multiple days, a resolver might only need to retrieve a fraction of the total zone for each update. Under such conditions, the traffic cost of maintaining a local root copy would differ substantially from the measurements reported in this study.

This idea does not reflect current root zone practice and is not implemented in existing distribution mechanisms. It has nevertheless been raised in technical discussions related to this work, including conversations with the project supervisor, as an example of how root zone production choices directly influence downstream traffic behavior. Any move in this direction would require changes at the level of root zone generation and signing, rather than adjustments at the resolver alone.

What such an approach would imply in practice depends on how modifications to the signing process interact with existing DNSSEC validation semantics and with the operational procedures currently used at the root. Any departure from the present all-at-once signing model would need to remain compatible with established trust chains and deployment expectations. These aspects are not visible from traffic measurements alone and cannot be evaluated using the experimental setup described in this work. As a result, this consideration lies outside the scope of the present analysis and is noted here as a structural factor that influences how local root traffic behaves.

Acknowledgements

I would like to thank my supervisor, dr.Marios Avgeris, for his continuous guidance and many helpful discussions throughout this project. I am also very grateful to Willem Toorop for his time, practical support, and for providing access to the experimental environment, which contributed greatly to the successful completion of this work.

References

[1] P.V. Mockapetris and K.J. Dunlap. "Development of the Domain Name System". In: *ACM SIGCOMM Computer Communication Review* 18.4 (Aug. 1988), pp. 123–133.

[2] P. Mockapetris. *Domain names - implementation and specification*. RFC 1035. Nov. 1987.

[3] Warren Kumari et al. *Making Local Root a Best Current Practice*. Internet-Draft draft-wkumari-dnsop-localroot-bcp. Work in progress. IETF, 2026.

[4] ICANN. *Reduced Risk of Redirected Query Traffic with Signed Root Name Server Data*. `https://www.icann.org/en/system/files/files/reduced-risk-redirected-query-traffic-signed-root-name-server-data-22may24-en.pdf`. Accessed: 2026-01-31. May 2024.

[5] Root Server System Advisory Committee. *RSSAC002: Measurements of the Root Server System*. `https://rssac002.root-servers.org/`. Accessed: 2026-01-12. 2023.

[6] Root Server System Advisory Committee. *RSSAC002 Measurements of the Root Server System*. `https://itp.cdn.icann.org/en/files/root-server-system-advisory-committee-rssac-publications/rssac-002-measurements-root-22jun23-en.pdf`. Accessed: 2026-01-12. 2023.

[7] *RFC 7706 Support in Knot Resolver*. Accessed January 2026. 2023. URL: `https://knot-resolver.readthedocs.io/en/v5.0.1/modules-rfc7706.html`.

[8] Warren Kumari and Paul Hoffman. *Running a Root Server Local to a Resolver*. RFC 8806. June 2020. URL: `https://www.rfc-editor.org/rfc/rfc8806`.

[9] Stephane Bortzmeyer, Ralph Dolmans, and Paul Hoffman. *DNS Query Name Minimisation to Improve Privacy*. RFC 9156. Feb. 2021. URL: `https://www.rfc-editor.org/rfc/rfc9156`.