HLOC: Hints-Based Geolocation Leveraging Multiple Measurement Frameworks

Quirin Scheitle, Oliver Gasser, Patrick Sattler, Georg Carle
Chair of Network Architectures and Services
Technical University of Munich (TUM)
Email: {scheitle,gasser,sattler,carle}@net.in.tum.de

***Abstract*—Geographically locating an IP address is of interest for many purposes. There are two major ways to obtain the location of an IP address: querying commercial databases or conducting latency measurements. For structural Internet nodes, such as routers, commercial databases are limited by low accuracy, while current measurement-based approaches overwhelm users with setup overhead and scalability issues.**

In this work we present our system HLOC, aiming to combine the ease of database use with the accuracy of latency measurements. We evaluate HLOC on a comprehensive router data set of 1.4M IPv4 and 183k IPv6 routers. HLOC first extracts location hints from rDNS names, and then conducts multi-tier latency measurements. Configuration complexity is minimized by using publicly available large-scale measurement frameworks such as RIPE Atlas. Using this measurement, we can confirm or disprove the location hints found in domain names. We publicly release HLOC's ready-to-use source code, enabling researchers to easily increase geolocation accuracy with minimum overhead.

I. Introduction

Geographical location of IP addresses has many purposes and users. Public service and businesses frequently focus on locating persons through their end-user devices. These human-centric use cases typically rely on easy-to-use commercial databases, which provide ever-increasing accuracy for edge nodes through feedback from users affected by wrong entries. For academia, the location of structural Internet nodes such as routers is elemental in studying and mapping the Internet. Unfortunately, commercial databases often provide low quality in this use case. State-of-the-art measurement-based algorithms are often not ready to use, but require implementation and setup of measurement infrastructure before use. For these reasons, structural Internet studies today frequently use commercial databases, and try to alleviate errors of these databases through laborious outlier analysis [9], [30], [32]. To cope with these shortcomings, we present HLOC, a framework combining the simplicity and scale of commercial geolocation databases with the accuracy of measurement-based approaches. The basic idea of HLOC is to leverage location hints frequently observed in router DNS names [7], [18]. As these hints may be incorrect or ambiguous [35], [42], HLOC interfaces with ready-to-use measurement frameworks to confirm or invalidate those hints through latency measurements. With the geographic coverage of frameworks such as RIPE Atlas [29], it is usually possible to find a probe very close to a location candidate and potentially confirm a location hint within a 100km radius. While a 100km radius would be considered city-level geolocation, HLOC may also return a bigger radius of confidence, which may be considered country-level geolocation. This radius of confidence can be configured by the user to match specific use cases. For the large scale of this work, which aims to locate all IPv4 and IPv6 router addresses found in CAIDA's traces, we precede this measurement step by integrating high-volume ZMap [8], [11] scans to (i) filter for responsive hosts and (ii) quickly locate the hemisphere of an IP address. This pre-measurement step can be skipped if only few IP addresses need to be located. We release HLOC source code and data used in this paper to the public. As RIPE Atlas automates data sharing between different measurements, the repeated usage of HLOC by multiple research teams will further improve its efficiency. Our main contributions in this work are:

- Creating HLOC, a ready-to-use geolocation framework leveraging DNS-based hint collection and distributed active measurements
- In-depth evaluation of HLOC geolocation measurements of 1.4M IPv4 and 183k IPv6 routers
- Comparison of HLOC geolocation results against commercial databases and previous measurement-based approaches

We structure this paper as follows: We discuss related work in Section II and present HLOC's architecture in Section III. We discuss measurement results in Section IV and evaluate our results against other approaches in Section V. We discuss limitations, trade-offs, future improvements and ethical considerations in Section VI. Section VII concludes this work.

II. Related Work

Related work on geolocation can be structured into three groups: measurement-based approaches, DNS-based approaches, and investigations on accuracy limitations of geolocation databases.

Measurement-based: There is a large body of measurement-based approaches for IP address geolocation. These approaches rely on latency, link level topology, time-to-live and other data to triangulate hosts or co-locate hosts to known landmarks [4], [14], [15], [17], [21], [23], [38]–[41].

DNS-based: In contrast to measurement-based approaches, DNS-based approaches leverage DNS data to derive an IP address's location. RFC 1876 defines a DNS extension capable of storing longitude/latitude data for IP addresses; however,

this extensions is rarely used, and would likely suffer from outdated information when IP addresses are being moved without updated DNS entries.

Several previous works use DNS data to locate IP addresses. Spring et al. [35] leverage DNS data to locate Internet routers for a topology study. Böttger et al. [3] study Netflix's infrastructure in depth by extracting information from DNS names and confirming those by measurements. Zhang et al. [42] show that misnamed router DNS names (e.g. due to re-assignment of IP addresses without changing the DNS name) can significantly distort topological studies. They suggest various rules to detect such misnaming, for example by disallowing location loops along a traceroute's path.

In 2014, Huffaker et al. [18] ("DRoP") investigate router DNS names and compare them against various ground truth domains and databases. They also use CAIDA's Internet measurement infrastructure to derive location rules for domains through latency and time-to-live analysis. While showing good results on few ground truth domains, DRoP does not provide a ready-to-use solution as its source code is not public, it leverages private measurement frameworks, and its generalization potential is unclear. In this work, we reappraise DRoP using our geolocation framework HLOC by (i) including more hint sources; (ii) using a more flexible search algorithm; (iii) leveraging public measurement frameworks; (iv) adding IPv6 capabilities and (v) publishing ready-to-use code and data.

Database Accuracy: Gueye et al. [13] in 2007 criticize the concept of geolocation databases to aggregate IP addresses into larger blocks. They show that some of these blocks can span 1,000km and more. Poese et al. [26] in 2011 investigate the accuracy of five different geolocation databases against ground truth data for one large Internet Service Provider. They find the databases to be partially incorrect on a city level, and heavily biased on a country level. They argue that this inconsistency is likely increased by block aggregation as criticized by Gueye et al. [13]. We strive to remediate accuracy problems of geolocation databases with HLOC by providing a ready-to-use geolocation framework with higher accuracy.

III. Architecture of HLOC

We design HLOC as a modular, flexible and extensible framework. This allows us to scale efficiently and accommodate changing requirements.

A. HLOC's Building Blocks

HLOC comprises of five building blocks: (i) parse codes, (ii) preprocess domains (iii) search codes in domains (iv) measure latency to hints, and (v) validate hints. Figure 1 shows these building blocks and their interfaces, which we describe in the following paragraphs.

Parse Codes: We gather several types of location codes, and map all possible *codes* of one *location* to a single location entry, amended by the location's coordinates and population data. We use four kinds of location codes: Airport codes [10] (example Houston, TX: IATA `HOU`, ICAO `KHOU`, FAA n/a), UN/Locode codes [37] (Houston: `US HOU`, transformed to `USHOU` for search), CLLI codes [36] (Houston: `HSTNTXMOCG0`, we use the first six characters `HSTNTX` as in [18]), and GeoNames city names (Houston: *Hiustonas*) [12]. The combination of those four kinds of locations codes results in 448k raw location codes. These codes do not necessarily provide the same GPS coordinates for a location, especially for cities with rather distant airports. We hence merge location codes in a radius of 100km around city centers to form one *location*. We use the number of inhabitants of a location to define a deterministic merge order, *i.e.,* smaller locations will be merged to bigger locations. This results in 5474 locations.

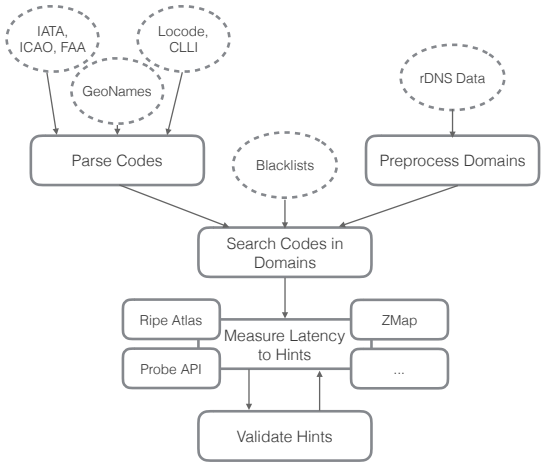

Fig. 1: The building blocks of HLOC.

Preprocess Domains: HLOC's input is a list of target IP addresses with corresponding DNS names. We use a reverse DNS file from Project Sonar [27] and filter it for routers using CAIDA's IPv4 ITDK [5] traces and IPv6 router names [6] datasets. We filter out ≈1% of invalid domains, for example containing characters not allowed in domains according to RFC 952 or top-level domains not existent according to IANA [19], for example, *.local*. For the remaining domains, we split DNS names into individual labels along "." boundaries and remove first and second level domain labels (cf. Section III-B).

Search Codes in Domains: With 448k location codes and millions of domain names in preliminary runs, an efficient data structure and search algorithm is key to our research. We find that organizing location codes in a prefix tree and then matching domains against this tree provides excellent performance, with about 10 minutes search time for our 1.6M domains (using 8 parallel processes). Figure 2 shows an excerpt of our location code prefix tree, with relevant codes for *Munich*. Matching a domain component *munich* against our location code prefix tree would result in intermediate and leaf node matches for *mun*, *munic*, *munich*. As we also match the subcomponents *unich*, *nich*, *ich*, the matches *uni*, *nic* and *nich* are also generated. Those matches of *domain labels* against the prefix tree of *location codes* are then named *location hints*.

Measure Latency to Hints: HLOC is designed to flexibly use multiple measurement frameworks. For our work, we

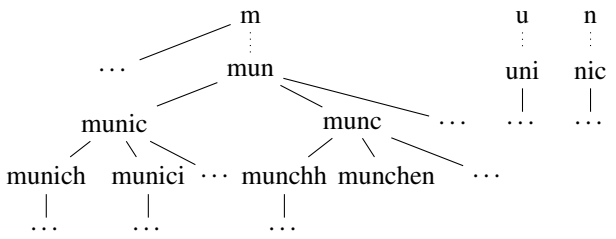

Fig. 2: Prefix tree excerpt with relevant codes for search term *munich*. Hierarchical layout for efficient search.

implement interfaces to three servers running ZMap and to the RIPE Atlas [29] measurement framework. Extensions to use, e.g., PlanetLab [25] (currently missing IPv6 support) or ProbeAPI [34], can be flexibly added. Queries and results are formed and processed in a unified format, which can easily be translated into framework-specific commands. Per domain name, we start measurements against one *location hint* and then call the *validate hints* block.

Validate Hints: The *validate hints* block is repeatedly called until all *location hints* for a domain were *falsified*, *i.e.,* excluded based on speed-of-light constraints, or one *location hint* was verified, *i.e.,* confirmed based on latency heuristics.

Specifically, HLOC's *validate hints* block processes the set of responsive IP addresses by repeatedly (i) excluding locations by latency constraints and (ii) verifying remaining hints using RIPE Atlas pin-point measurements.
Validation through pin-point measurements requires (a) finding a probe p nearby the *location hint* h (cf. Equation 1)and (b) latency constraints (cf. Equation 2) holding true.

$$d(p,h) < x \tag{1}$$

$$RTT(p,h) < a + \frac{2 \cdot d(p,h)}{c \cdot c_0} \tag{2}$$

Equation 1 limits the maximum distance of a probe p to a location hint h. For our study, we set x to 1,000km, and highlight that HLOC will still choose the closest probe possible, but remote locations with sparse coverage benefit from a larger x. We discuss the impact of x in Section IV-B.

In Equation 2, $RTT(p,h)$ is the measured round trip time between probe p and hint location h, a is a latency buffer, $d(p,h)$ is the geographical distance between probe and hint, c_0 the speed of light in vacuum, and c the inverse refractive index for fiber. We use c=⅔ [33].

We argue that the latency buffer a is necessary to accommodate for time a packet can spend in buffers or scheduling on shared measurement platforms. Based on initial measurement results and our measurement targets, we consider a=9ms as a good starting value. Studies requiring higher precision can adjust this parameter. We conduct a sensitivity analysis and discussion of this parameter choice in Section IV-B.

These settings result in a maximum possible error in our studies of $2x + a_{km}$ =∼2,900km, with $a_{km} = a \cdot c \cdot c_0 \cdot 1/2$.

B. Balancing Efficiency and Accuracy

We resolve various challenges through iterative improvements, usually trading magnitudes of efficiency gain for minor potential losses in accuracy:

Challenge 1 - Abundance of Matches: Initial runs returned on average >20 matches per DNS name, resulting in an infeasible amount of matches. We contribute several approaches to reduce matches while not significantly reducing accuracy.

First, we focus on larger locations by removing locations below a certain inhabitant threshold. For this work, we choose a threshold of 100,000 inhabitants. Again, other studies can easily chose a different threshold, and we conduct sensitivity analysis and discussion for this threshold in Section VI-E. This reduces the average number of matches per IP address from >20 to ≈7. It is possible to white-list important smaller locations (such as fiber landing points).

Second, we apply various blacklisting techniques to remove dominant false matches:

- 26 codes such as *tel* (part of *telecom*) or *cpe* (*customer premises equipment*)
- 425 words not to be searched for location hints, *e.g., Internet*, *Linux* or *static*
- 35 code-location mappings considered wrong. For example, *lin* is the IATA code for Milan, but also matches *Illinois, Carolina* and *Dublin*. In this example, the code-location blacklist will forbid *lin* to match *Illinois, Carolina* and *Dublin*.

We create these blacklists through iteratively assessing the top matches, thus focusing on high-impact false matches. This manual step may seem inefficient, but its one-off character and high-impact focus make it well feasible. By filtering the 486 aforementioned values, we reduce the number of matches from ≈7 to ≈4.4. We thoroughly document our blacklist with examples to enable community review and contribution.

Third, we remove matches for top- and second-level domains. These matches typically represent the headquarters' location (*e.g., .fr*) or the company name (*e.g., vivendi.fr*), not specific router locations. This brings the number of matches per IP address further down from ≈4.4 to ≈3.9.

Another option is to remove rDNS names that embed the IP address. These are typically automatically generated without specific location hints. We support a variety of IPv4 and IPv6 embedding schemes. In this work, we separately investigate DNS names with and without embedded IP addresses.

Through these measures we reduce the average number of matches per DNS name from >20 to ≈3.9. Please note that this effect is amplified by fully removing IP addresses with no matches left, leading to a similar reduction of total matches.

Challenge 2 - Validation Runtime: Measurement frameworks such as RIPE Atlas allow a certain number of measurements per time interval, limiting the number of queries we can conduct. Therefore, reducing the number of required measurements is our main lever to reduce validation runtime: First, we precede the wide-spread measurements by conducting ZMap measurements from servers in Dallas, Frankfurt and

Singapore, which allows speedy filtering of unresponsive IP addresses. Also, the three geographically spread out servers can often locate the hemisphere of an IP address's location, excluding further matches. This step brings the number of possible matches down from ≈3.9 to ≈1.3, and at the same time excludes unresponsive IP addresses.

IV. Measurement Results

In this section we present the results of our measurements in terms of verified and falsified geolocation hints, differences between IP-encoding and not IP-encoding domain names, and contribution of each of our four location code types.

A. Measurement Statistics

For our large-scale evaluation, we source IPv4 and IPv6 router IP addresses from CAIDA's May 2016 ITDK/DNS names datasets [5], [6]. From these, we filter out ≈1% of invalid domains (see Section III-A). The remaining domain names are then matched against the location hints in our prefix tree. 41% (IPv4) and 4% (IPv6) of IP addresses do not produce any location matches and are hence not further processed. Some examples for such DNS names without matches are: *rf-rtr01.ew.net.nz*, *host.3.static.cardbankph.com*, *rt230bb131-145-61.routit.net* and *internet-gw.customer.alter.net*. Table I summarizes statistics for the aforementioned steps. The set of IP addresses with matches (58%/96%) is then processed in the next steps of our algorithm.

We now describe the location measurement phase, with its statistics displayed in Table II. We first filter 2% of IP addresses from either our blacklist (curated from previous studies) or not announced to our BGP border router.

Next, we conduct ZMap *ICMP echo request* latency measurements and filter out unresponsive IP addresses. These high-volume measurements eliminate 28% of IPv4 addresses and 78% of IPv6 addresses. The value for IPv4 is in the expected range, given that our dataset also includes access and home routers. The 78% of non-responsive IPv6 addresses are surprising, however we find the total number of 29k responsive IPv6 routers roughly in line with previous studies (35k in [2], 43k in [11]). Closer analysis reveals that a very large part (75%) of our IPv6 dataset consists of dynamically generated IPv6 DNS names of the form *node-1w7jr9y4otrxqsxabcdek4c1.ipv6.telus.net*. This subset offers a response rate of only 0.4%, which is not surprising as the addresses typically contained the SLAAC-characteristic `ff:fe` middle bytes. These are therefore most likely home routers which could block ICMP requests. Please note that the non-IP-encoded IPv6 subset offers a significantly higher ICMP response rate of 73%.

Before discussing the breakdown of measured and responsive IP addresses into the three categories of *verified hint*, *no verified hint*, and *all hints falsified*, we conduct a sensitivity analysis parameters of our algorithm.

B. Sensitivity Analysis and Error Margins

We consider a *location hint* as *verified* if it satisfies Equations 1 and 2. These equations contain margins for probe distance and latency. These margins are required, as finding a probe with zero distance to a location hint is as unlikely as a latency measurement equal to the direct distance. Margin calibration strikes a balance between false positive and false negative rates. Before discussing the results of our algorithm, we investigate the sensitivity of our results to these margins.

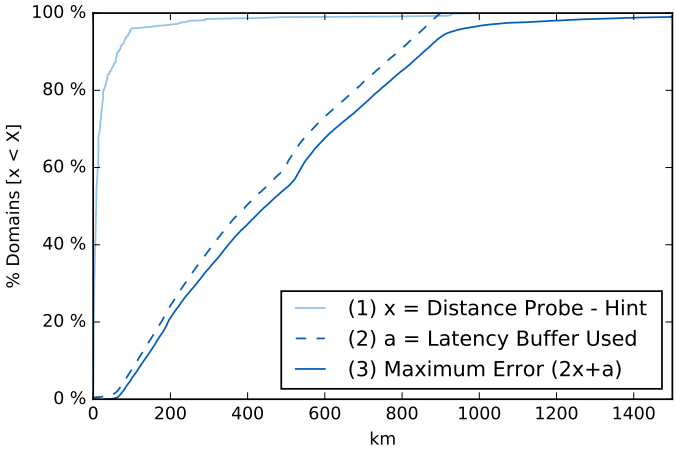

Fig. 3: Verified location hints: CDF of (1) distance between probe and location hint – 80% under 25km, (2) Amount of latency buffer used - linear increase towards 9ms threshold, and (3) linear increase in maximum possible error, 93% below 900km. IPv4 and IPv6 almost identical, hence mean shown.

Figure 3 investigates probe distance and maximum error for verified location hints. We see that probe distance is typically very small, with 80% below 25km. We also see that the latency buffer used, which we set to a maximum of 9ms, is rising linearly (*i.e.,* the underlying distribution is uniform in that interval). The maximum error, as a function of both, also raises linearly to a point of 93% of IP addresses at 900km. We find the 7% of high-error outliers to be typically caused by high probe distances. One such example is Summer Beaver, a remote airport in Canada, with the next RIPE Atlas probe being 927km and a minimum of 16ms away.

We next analyze the influence that our parameter choice has on not verified hints: Figure 4 analyzes the excessive latency of not verified hints, *i.e.,* how much a certain latency did exceed our thresholds for confirmation of a hint. We see the excessive latency to continue the linear rise see in Figure 3 up to a point of ∼50ms, with slightly better values for IPv6.

We conclude that, as the latency buffer used rises linearly until and beyond (cf. Figure 4) 9ms, our selected margin of 9ms offers linear sensitivity to the number of verified hints and may easily be adapted by other studies to smaller or larger values. For probe distance, permitting a maximum distance of 100km will permit to obtain 93% of results without error margins rising beyond linear. However, this would come at the cost of not being able to verify remote locations at all.

We next analyze the numbers of verified and unverified hints that result from applying our algorithm.

TABLE I: Statistics for Location Hints Matching.

# IP addresses	IPv4			IPv6		
Processing Step	Total	IP encoded	IP not encoded	Total	IP encoded	IP not encoded
Routers	2.5M (100%)	1.4M (100%)	1.0M (100%)	190k (100%)	146k (100%)	44k (100%)
Invalid Domains	-14k (0.6%)	breakdown n/a		-0.3k (0.1%)	breakdown n/a	
No Match	-1.0M (41.3%)	-821k (58.2%)	-192k (18.6%)	-7.2k (3.8%)	-2.7k (1.8%)	-4.5k (10.1%)
Remaining	1.4M (58.2%)	590k (41.6%)	836k (81.6%)	183k (96.0%	143k (98.2%)	39k (89.6%)

TABLE II: Statistics for measurement algorithm: ZMap pre-scan effective for detection of unresponsive hosts, DNS names with no encoded IP addresses offer better results.

Processing Step	IPv4			IPv6		
	Total	IP encoded	IP not encoded	Total	IP encoded	IP not encoded
Input IP addresses	1,426k (100%)	590k (100%)	836k (100%)	183k (100%)	**143k** (100%)	39k (100%)
- Filtered IP addresses[1]	-34k (2.4%)	-7k (1.2%)	-27k (3.3%)	-2k (1.3%)	-.2k (0.2%)	-2k (5.2%)
- ZMap/censys timeouts	-391k (27.5%)	-217k (36.8%)	-174k (20.8%)	-143k (78.4%)	-139k (97.1%)	-4k (10.1%)
- RIPE timeout	-40k (2.8%)	-22k (3.7%)	-18k (2.1%)	-8k (4.3%)	-3k (2.3%)	-5k (11.8%)
Responsive	961k (67.4%)	344k (58.3%)	617k (73.8%)	29k (16.0%)	1k (0.4%)	29k (73.0%)
Responsive	961k (100%)	344k (100%)	617k (100%)	29k (100%)	1k (100%)	29k (100%)
All hints falsified[2]	417k (**43.4%**)	132k (38.3%)	285k (46.2%)	7k (22.9%)	.3k (47.3%)	6k (22.4%)
Hint verified	**45k** (4.7%)	**10k** (2.8%)	**35k** (5.7%)	**5k** (17.6%)	**8** (1.3%)	**5k** (18.0%)
No hint verified	500k (52.0%)	203k (59.0%)	297k (48.1%)	17k (59.5%)	.3k (51.4%)	17k (59.7%)
Without probe	17k (1.8%)	5k (1.5%)	12k (2.0%)	1k (4.2%)	.1k (12.2%)	1k (4.4%)
Latency too high	482k (50.1%)	197k (57.4%)	284k (46.1%)	16k (54.9%)	.2k (39.3%)	16k (55.2%)

1: Blacklisted or not announced IP addresses 2: About 1% falsified by ZMap/censys.

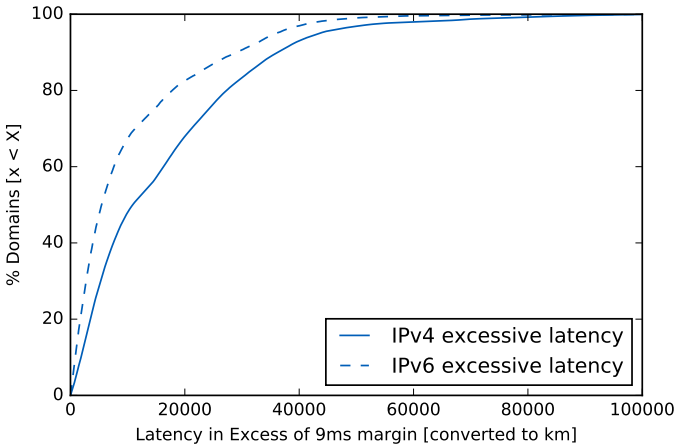

Fig. 4: Not verified location hints: Excessive Latency (round trip time minus the verification threshold) raises linearly before dropping off into a long tail. IPv6 slightly better.

C. Verified Hints

We can verify a location hint for 4.7% of responsive IPv4 (corresponds to 45k addresses) and 17.6% of responsive IPv6 addresses (5k addresses). While 4.7% and 17.6% are rather small portions of all initial IP addresses, we argue that 45k (IPv4) and 5k (IPv6) of successfully located router IP addresses prove that the concept of extracting DNS hints and validating them through RIPE Atlas works for a significant number of routers. We also find that the routers with a validated location play a central role in the Internet's topology: Validated IP addresses were on average present in 12,585 of CAIDA's May 2016 path measurements compared to 3,567 traces for not validated IP addresses. We argue that this more central role likely comes with higher uptime and response rates and lower latency, hence allowing for higher validation rates. We discuss avenues to further raise the number of validated IP addresses in Section VI.

D. No Verified Hint

We consider a domain as *no verified hint* if we can neither verify any hint nor falsify all hints for a domain. At about 55%, the majority of responsive IP addresses falls into this category of *no verified hint*. One small root cause for this is the lack of a nearby probe, which amounts to 2% (IPv4) and 4% (IPv6). This number of location hints without a nearby probe might be reduced through the adaption of further measurement frameworks and increased geographic coverage of RIPE Atlas. The majority of unverified IP addresses is due to high latency. We discuss mitigation strategies for this in Section VI.

E. All Hints Falsified

If all *location hints* for a DNS name were falsified based on speed-of-light constraints, we label that DNS name with the results *all hints falsified*. With 43% (IPv4) and 23% (IPv6) of responsive IP addresses, this category has a surprisingly high share of all responsive IP addresses. We manually investigate a sample of this group and find it to usually be caused by matches created on strings such as company names, which do not necessarily express an individual IP addresses location.

F. IP-Encoding Domain Names

This section compares the group of IP-encoding DNS names with that of not IP-encoding DNS names. A common hypothesis is that IP-encoding DNS names are automatically generated, implying fewer or less accurate geographic hints.

Tables I and II split out this subgroup for comparison. We find DNS names with encoded IP addresses to represent a significant share of IPv4 (41%) and IPv6 (78%) router IP addresses. Those DNS names that encode IP addresses generate fewer location matches, respond less likely to *ICMP echo requests* and less likely have their location hints validated.

However, 10k IPv4 addresses from the IP-encoding provide verified location hints. We further analyze this verified subgroup IP-encoding DNS names, and commonly find city or regional names in labels above the IP-encoded label. Table III displays examples of this behavior.

TABLE III: Examples of verified IP-encoding DNS names.

DNS Name	Location	Code	RTT
*1-2-3-4.lightspeed.**hstntx**.sbcglobal.net*	Houston, TX	CLLI	3.5ms
*1-2-3-4.lightspeed.**miamfl**.sbcglobal.net.*	Miami, FL	CLLI	4.2ms
*ip-1-2-3-4.**mel**.xi.com.au.*	Melbourne (AU)	IATA	6.0ms
*rrcs-1-2-3-4.**nyc**.biz.rr.co.*	NYC, NY	IATA	3.7ms

G. Contributions per Location Hint Source

We next evaluate the contribution of various location sources to the number of generated and verified location hints. Table IV shows location hint sources along their numbers of codes, generated hints, and verified hints. Over 99% of verified matches stem from IATA, Geonames and CLLI codes, while ICAO, FAA and UN/LOCODE codes provide very few verified hints. GeoNames and CLLI matches are very efficient when comparing number of hints to number of verified hints. As CLLI and GeoNames codes are longer than, *e.g.,* IATA's 3-letter codes, they are less prone to matching strings by chance. We discuss in Section VI how these insights can potentially improve HLOC by weighting location hint sources.

TABLE IV: IATA, GeoNames and CLLI codes provide 99% of verified hints.

Category	IATA	ICAO	FAA	UN/LO	GeoNames	CLLI
# Codes	8k	13k	20k	77k	32k	31k
Hints (100%)	4.5M	209k	472k	59k	215k	167k
Verified	**32k**	122	413	120	**13k**	**5k**
Verified (%)	.7%	$<$ .0%	.1%	$<$ .0%	**5.9%**	**2.8%**

V. Evaluation

In this section we compare the locations obtained from our HLOC geolocation framework to (i) commercial geolocation databases and (ii) to Huffaker et al.'s results obtained by using their DRoP system [18]. Table V details this comparison.

A. Comparison to commercial databases

We compare our results against MaxMind's [22] and ip2location's city-level geolocation databases [20]. We report findings in three possible categories: Based on HLOC's hints and measurements, a database entry may either be the *same*, *i.e.,* HLOC could verify a location hint and the database reports the same location, *possible*, *i.e.,* the database location is different from HLOC, but possible based on HLOC's latency measurements, or *wrong*, *i.e.,* the location reported by a database has been falsified based on HLOC's latency measurements and speed-of-light constraints.

HLOC Verified IPv4: We first look at IPv4 locations that have been verified by HLOC. Looking at the *verified (IPv4)* row in Table V, we see that, based on our measurements, 44% of locations reported by GeoLite are wrong, *i.e.,* falsified through measurement, compared to about 12% from ip2location. This confirms that commercial databases, when applied to router datasets, contain a non-negligible number of incorrect entries. In addition, we can compare the percentage of cases where a verified HLOC location is the same as the city reported by a database. We highlight that as neither the databases nor HLOC form a definitive ground truth, the *same* category is rather indicative. However, the difference of 77% agreement for ip2location and 40% for GeoLite is noteworthy. **HLOC Unverified IPv4:** For location hints that HLOC can not verify, HLOC's latency measurements still disprove a small number of locations returned by commercial databases.

IPv6: Table V displays IPv6 results where they significantly differ from IPv4 results. GeoLite does not support IPv6, and ip2location IPv6 matches are significantly more often wrong than IPv4 matches (64% IPv6; 12% IPv4). This is likely linked to the large IPv6 address space, pressuring database providers to aggressively group IP addresses into the blocks mentioned in Section II. This highlights the need of contrasting database information with latency-based measurements for IPv6.

B. Comparison to DRoP's ruleset

DRoP does not provide a geolocation for 84% of IPv4 router addresses. This is caused by DRoP's operating mechanism based on specific regular expression rules for a set of specific domains. However, for those IP addresses where DRoP does deliver an answer, the answer interestingly is typically verified or falsified, and only very rarely in the "possible" range. We highlight that although the DRoP rules are very narrow, specifying a specific kind of code (for example, IATA) at a specific location in a specific domain, its answers are still frequently falsified through latency measurements. This confirms the work of Zhang et al. [42] stating that DNS names are often outdated or wrong. It also confirms that DNS-based location approaches should be combined with latency measurements to minimize the influence of wrongly named IP addresses. Interestingly, the DRoP results are much better for IPv6 - this might be due to less pressure to re-use and re-locate addresses in the vast IPv6 address space.

TABLE V: Evaluation of location decisions by databases and DRoP against HLOC measurements: ip2location more accurate than GeoLite, DRoP frequently with "no data". All information-based approaches with a significant number of wrong decisions.

	HLOC		GeoLite			ip2location			DRoP			
	Location Dec.	n	Same	Possible	Wrong	Same	Poss.	Wrong	Same	Poss.	Wrong	No data
IPv4	Verified	45k	40.4%	15.6%	**44.0%**	**76.6%**	11.3%	**12.1%**	7.8%	0.1%	8.4%	**83.7%**
	All falsified	417k	n/a[1]	100%	0%	n/a	100%	0%	n/a	n/a	2.2%	97.8%
	No verified	499k	n/a	96.1%	3.9%	n/a	98.8%	1.2%	n/a	10.5%	4.1%	85.4%
	Timeout	465k	n/a	100%	n/a[2]	n/a	100%	n/a	n/a	26.4%	n/a	73.6%
IPv6	Verified	5k		—		25.7%	10.6%	**63.6%**	33.7%	1.0%	1.8%	63.5%
	No verified	17k		—		n/a	74.2%	**23.9%**	n/a	25.5%	3.3%	71.2%

1: With no verified HLOC match, other approaches can not have the same match. 2: With HLOC timeout, it is not possible to evaluate other approaches.

TABLE VI: For DRoP's ground truth domains, we show performance for (a) DRoP's reported performance, (b) our reproduction of DRoP and its validation against latency measurements, and (c) HLOC-generated hints and their latency validation.

Domain	DRoP 2014 [18]				DRoP 2016 Reproduction					HLOC			
	n	Type	Match	TP[1]	n	Match	TP[1]	Ver.[2]	Fals.[2]	Match	TP[1]	Ver.[2]	Fals.[2]
belwue.de	161	City	52%	86%	53	64%	65%	22	1	94%	64%	32	5
cogentco.com	13,129	IATA	90%	99%	9,475	95%	26%	2,381	**628**	99%	23%	2,144	295
digitalwest.net	111	IATA	49%	100%	47	49%	26%	6	0	100%	15%	7	2
ntt.net	2,584	CLLI	96%	100%	3,125	54%	37%	**622**	5	99%	30%	**937**	148
peak10.net	115	IATA	100%	100%	199	99%	9%	18	0	100%	9%	18	0

1: % of matches that are true positives 2: Total count of verified or falsified matches. "possible" and "time out" results not displayed.

C. Comparison to DRoP ground truth domains

In this section, we compare the accuracy of HLOC and DRoP against the set of 5 ground truth domains from DRoP. For those domains, the authors of DRoP had confirmed the accuracy of their rule set.

Table VI first shows the original results of Huffaker et al. [18]. As they do not provide their dataset, we reproduce their results for a host-by-host comparison, which we show next. Please note the different number of hosts n, as we filter for all routers in a domain (according to CAIDA traces), while DRoP authors likely picked the subset of hosts that they obtained a ground truth for. Also, we limit this comparison to IPv4 as original DRoP numbers do not include IPv6. We reproduce DRoP results by applying their published matching rules and run HLOC's measurement algorithm to verify or falsify their results. We also give the results for matches generated and measured by HLOC. Our reproduction, though working on a different numeric base, typically achieves similar match rates. The difference between the original DRoP result and our reproduction of *ntt.net* is likely caused by NTT's creation of international CLLI codes, which are not part of the standard and are hence missing in our dataset. The DRoP team had likely created these easily decipherable codes such as *londen*, *taiptw*, *newthk* and *amstnl* by hand. Using HLOC, we can typically verify a large part of the DRoP reproduction matches, but for *cogentco.com* can also falsify over 600 matches. The last aggregate column are the results of HLOC.

The application of HLOC's full matching tree generates more matches than the DRoP rules, with the count of verified matches frequently equaling or surpassing the number of verified matches from DRoP's rules. We investigate those cases where HLOC can verify matches that the DRoP rules can not find. We find no systematic pattern, but incidences of (i) manually named DNS records not in accordance with DRoP rules and (ii) finding (and verifying) matches in DNS names that seem not to intend to give a location hint. Based on our reproduction of DRoP, we argue that HLOC's generic matching equals or surpasses DRoP's specific domain patterns just two years after their creation. We also highlight that even specific DRoP rules can be falsified for a significant number of hosts. This confirms the unavoidable existence of outdated and misnamed DNS names, requiring latency based verification/falsification as done by HLOC.

VI. Discussion and Limitations

We discuss our work, its current limitations, and ways to tackle these in this section.

Throughout this work, we have confirmed that for router datasets, geolocation database results have a significant number of wrong entries and should be contrasted by latency measurements, an issue even worse with IPv6. We show that leveraging ready-to-use public measurement frameworks in HLOC works well, and that a significant number of *location hints* can be confirmed with singular latency measurements. Using few and simple measurements is an important scale factor when leveraging public frameworks that often charge by amount and complexity of measurements. We find the error margin to increase linearly for a very large fraction of measurements, easily enabling users of HLOC to set an individually acceptable error margin.

We now discuss HLOC's current design choices and plans for improved future versions.

A. Measurement Scope and Nature

Since Internet-wide geolocation measurements from hundreds of distributed probes are infeasible, we limit HLOC's

scope to geolocating routers. This also ensures that we adhere to RIPE Atlas API thresholds, since HLOC is measurement-bound and not computationally-bound. In the future we want to perform multiple measurements for one router over time to avoid temporary congestion and queuing situations.

B. Probe Selection

The current version of HLOC randomly selects a RIPE Atlas probe close to a location hint. This offers various ways for improvement: First, RIPE Atlas probes with high first- or second-hop latencies could be removed. As each RIPE Atlas probe continuously measures first- and second-hop latency, probes behind high-latency links can be excluded from our measurements. Second, selection of probes within the same Autonomous System as the target IP address can avoid high latencies due to geographically sparse inter-AS connections. Third, we could select several suitable probes, aiming to increase the probability of hitting a low-latency measurement caused by an empty link or a low probe workload. Holterbach et al. [16] show that measurement interference can significantly bias RIPE Atlas latency measurements. This step increases the measurement cost and might not be feasible for some studies. Fourth, consensus-based blacklisting can remove probes with wrong locations. Fifth, the integration of further frameworks can aid probe sparsity and probe diversity.

C. Probe Distance, Latency Buffer, and Error Margin

As discussed in Section IV-B, our parameters for probe distance and latency buffer offer largely linear sensitivity and can easily be adjusted to suit different requirements. A smart way to combine both parameters will be for the user to set a maximum acceptable error margin, towards which HLOC will optimize by dispatching more measurements for extreme cases. Future versions may also show possible location areas of an IP address, however these will be of complex shape.

D. Location Codes Ordinance and Search

Currently, HLOC probes matches for a domain in the order in which they are found in our code prefix tree. Future versions should leverage smart sorting to reduce the average number of measurements needed to come to a conclusion for a domain. Supported by our results from Section IV-G, this could include sorting by match length (putting higher weight on longer matches), by confirmation count (*i.e.,* measuring frequently correct hints first), or by the location of a match within a label (e.g., prioritizing isolated matches over matches within a word). Reinforcement learning to determine typical code types and locations per domain [18] may also help.

E. Location Size Threshold

The current version of HLOC uses a location size threshold of 100k inhabitants: Only locations above this threshold will be able to generate matches. This threshold is mainly applied to reduce the abundance of matches generated per domain. As discussed in Section III, this threshold can both be changed for its absolute value as well as for individual exceptions.

We conduct a sensitivity analysis on this threshold value, evaluating the average number of matches per domain after exclusion of locations based on ZMap latency measurements.

Using a threshold of 100k inhabitants yields 7.7 average matches per domain (cf. Section III). For a threshold of 50k, this rises to 11.34, and for a threshold of 1k, this number rises to even 20.44. As our measurements already exceed many of RIPE Atlas's limitations for a threshold of 100k, we conclude that 100k was the correct threshold to choose for our large-scale measurements. Typical path analysis use cases [30], [32], only locate thousands instead of millions of IP addresses and could easily use a lower threshold. Also, the steps discussed in Section VI-D could likely reduce the number of required measurements for domains with many matches.

F. Measuring IP Anycast Addresses

IP anycast routes packets for one IP address to one of several distinct nodes, usually in separate geographical locations, used for example by DNS root servers [28]. When performing latency measurements, HLOC selects a probe geographically close to each hint. As HLOC, in its current version, stops after having validated a first hint, it would disprove other valid locations for multicast IP addresses. This has a small impact on our results, as (i) IP anycast addresses typically do not encode locations in their DNS name; and (ii) we are not aware of IP anycast being used for Internet routers to a significant extent.

G. Ethical Considerations

We follow an internal multi-party approval process, among others based on Partridge and Allman [24], before any measurement activities are carried out. We conclude that our *icmp echo request* measurements consisting of a few packets per week per targeted router, and the resulting data, can not harm individuals, but may result in investigative effort for system administrators. We aim to minimize this effort by deploying scanning best-practice efforts of (i) using dedicated scan machines with explanatory websites, (ii) maintaining a blacklist, (iii) replying to every abuse e-mail (none received in this experiment), and (iv) minimizing impact on RIPE Atlas in coordination with the RIPE Atlas team.

VII. Conclusion

We present HLOC, a framework that derives geolocation hints from DNS names and uses publicly available measurement frameworks to validate or falsify theses hints through latency measurements. We evaluate its performance on IPv4 and IPv6 router datasets and verify 45k IPv4 addresses and 5k IPv6 addresses. We compare our results to those from geolocation databases and DRoP and find that we can frequently disprove those base on latency constraints. We provide HLOC as ready-to-use tool for other research groups.

Data and Code Release: In [31], we outline our aim for repeatable, replicable and reproducible research as defined by ACM [1], and publish HLOC code and data on GitHub:

`https://github.com/tumi8/hloc`

This repository will also be used for future development.

Acknowledgments: We thank Johannes Naab for his input, the Leibniz Supercomputing Centre (LRZ) of the Bavarian Academy of Sciences (BAdW) for cloud computing infrastructure, and the RIPE Atlas team for their support. This work has been supported by the German Federal Ministry of Education and Research, project X-CHECK, grant 16KIS0530, and project AutoMon, grant 16KIS0411.

REFERENCES

[1] Association for Computing Machinery. Result and Artifact Review and Badging. http://acm.org/publications/policies/artifact-review-badging, Acc. Jan. 20, 2017.

[2] R. Beverly, M. Luckie, L. Mosley, and K. Claffy. Measuring and Characterizing IPv6 Router Availability. In *Passive and Active Measurement*, 2015.

[3] T. Böttger, F. Cuadrado, G. Tyson, I. Castro, and S. Uhlig. Open Connect Everywhere: A Glimpse at the Internet Ecosystem through the Lens of the Netflix CDN. *arXiv*, 2016.

[4] M. Calder, X. Fan, Z. Hu, E. Katz-Bassett, J. Heidemann, and R. Govindan. Mapping the expansion of Google's serving infrastructure. In *ACM SIGCOMM Conference on Internet Measurement*, 2013.

[5] Center for Applied Internet Data Analysis. Internet Topology Data Kit. http://http://www.caida.org/data/internet-topology-data-kit/, Acc. June. 3, 2016.

[6] Center for Applied Internet Data Analysis. IPv6 DNS Names Dataset. http://www.caida.org/data/active/ipv6_dnsnames_dataset.xml, Acc. Sep. 29, 2016.

[7] J. Chabarek and P. Barford. What's in a Name? Decoding Router Interface Names. In *ACM Workshop on HotPlanet*, 2013.

[8] Z. Durumeric, E. Wustrow, and J. A. Halderman. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*, Washington, D.C., 2013. USENIX.

[9] A. Edmundson, R. Ensafi, N. Feamster, and J. Rexford. Characterizing and Avoiding Routing Detours Through Surveillance States. *arXiv*, 2016.

[10] Fubra Ltd. World Airport Codes. www.world-airport-codes.com, Acc. Aug 3, 2016.

[11] O. Gasser, Q. Scheitle, S. Gebhard, and G. Carle. Scanning the IPv6 Internet: Towards a Comprehensive Hitlist. In *Traffic Monitoring and Analysis*. IFIP, 2016.

[12] GeoNames. GeoNames Codes. www.geonames.org, Acc. Aug 1, 2016.

[13] B. Gueye, S. Uhlig, and S. Fdida. Investigating the Imprecision of IP Block-Based Geolocation. In *Passive and Active Measurement*, 2007.

[14] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida. Constraint-Based Geolocation of Internet Hosts. *IEEE/ACM Transactions On Networking*, 2006.

[15] C. Guo, Y. Liu, W. Shen, H. J. Wang, Q. Yu, and Y. Zhang. Mining the Web and the Internet for Accurate IP Address Geolocations. In *INFOCOM*, 2009.

[16] T. Holterbach, C. Pelsser, R. Bush, and L. Vanbever. Quantifying Interference between Measurements on the RIPE Atlas Platform. In *ACM SIGCOMM Conference on Internet Measurement*, 2015.

[17] Z. Hu, J. Heidemann, and Y. Pradkin. Towards Geolocation of Millions of IP Addresses. In *ACM SIGCOMM Conference on Internet Measurement*, 2012.

[18] B. Huffaker, M. Fomenkov, and k. c. Claffy. DRoP: DNS-Based Router Positioning. *ACM SIGCOMM Computer Communication Review*, 2014.

[19] IANA. TLD List. www.iana.org/domains/root/db, Acc Sep. 29, 2016.

[20] ip2location. City database. http://www.ip2location.com/databases/db5, Acc. Oct. 7, 2016.

[21] E. Katz-Bassett et al. Towards IP Geolocation Using Delay and Topology Measurements. In *ACM SIGCOMM Conference on Internet Measurement*, 2006.

[22] MaxMind. Geolite2 city. http://dev.maxmind.com/geoip/, Acc. Sep. 25, 2016.

[23] V. N. Padmanabhan and L. Subramanian. An Investigation of Geographic Mapping Techniques for Internet Hosts. In *ACM SIGCOMM Computer Communication Review*. ACM, 2001.

[24] C. Partridge and M. Allman. Ethical Considerations in Network Measurement Papers. *Communications of the ACM*, 2016.

[25] PlanetLab. http://planet-lab.org/, Acc. Sep 29, 2016.

[26] I. Poese, S. Uhlig, M. A. Kaafar, B. Donnet, and B. Gueye. IP Geolocation Databases: Unreliable? *ACM SIGCOMM Computer Communication Review*, 2011.

[27] Rapid7. Project Sonar Reverse DNS. https://scans.io/study/sonar.rdns, Acc. Aug 1, 2016.

[28] RIPE NCC. DNS Root Servers. http://root-servers.org/index.html, Acc. Oct. 10, 2016.

[29] RIPE NCC. RIPE Atlas. https://atlas.ripe.net/, Accessed September 29, 2016.

[30] Q. Scheitle, M. Wachs, J. Zirngibl, and G. Carle. Analyzing Locality of Mobile Messaging Traffic Using the MATAdOR Framework. In *Passive and Active Measurement*, 2016.

[31] Q. Scheitle, M. Wählisch, O. Gasser, T. C. Schmidt, and G. Carle. Towards an Ecosystem for Reproducible Research in Computer Networking. In *ACM SIGCOMM 2017 Reproducibility Workshop*.

[32] P. Schmitt, M. Vigil, and E. Belding. A Study of MVNO Data Paths and Performance. In *Passive and Active Measurement*, 2016.

[33] A. W. Snyder and J. Love. *Optical Waveguide Theory*. Springer, 2012.

[34] Speedchecker Ltd. ProbeAPI. http://probeapi.speedchecker.xyz/, Acc. Sep 29, 2016.

[35] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP Topologies with Rocketfuel. In *ACM SIGCOMM Computer Communication Review*, 2002.

[36] Telcodata. CLLI codes. www.telcodata.us, Acc. Aug 1, 2016.

[37] UN. LOCODE. www.unece.org/cefact/codesfortrade/codes_index.html, Acc. Aug 1, 2016.

[38] Y. Wang, D. Burgener, M. Flores, A. Kuzmanovic, and C. Huang. Towards Street-Level Client-Independent IP Geolocation. In *NSDI*, 2011.

[39] B. Wong, I. Stoyanov, and E. G. Sirer. Octant: A Comprehensive Framework for the Geolocalization of Internet Hosts. In *NSDI*, 2007.

[40] K. Yoshida et al. Inferring PoP-level ISP Topology through End-to-End Delay Measurement. In *Passive and Active Measurement*, 2009.

[41] I. Youn, B. L. Mark, and D. Richards. Statistical Geolocation of Internet Hosts. In *International Conference on Computer Communications and Networks*. IEEE, 2009.

[42] M. Zhang, Y. Ruan, V. S. Pai, and J. Rexford. How DNS Misnaming Distorts Internet Topology Mapping. In *USENIX ATC*, 2006.