Multi-Layered Telemetry Assessing Global Performance of LEO Internet Providers

Enhancing LEO Internet Providers Telemetry with User-Initiated Active Measurements

Janusz Urbański[1]

Supervisor(s): Dr. Nitinder Mohan[1]**, Dr. Tanya Shreedhar**[1]

[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2025

Name of the student: Janusz Urbański
Final project course: CSE3000 Research Project
Thesis committee: Dr. Nitinder Mohan, Dr. Tanya Shreedhar, Dr. Qing Wang

Abstract

Low Earth Orbit (LEO) satellite constellations, particularly SpaceX's Starlink, have quickly gained popularity and have become a viable alternative to traditional terrestrial Internet Service Providers (ISPs) in recent years. However, due to their novelty and unique architecture, research into their performance is limited, especially one comparing LEO and terrestrial internet. This paper will demonstrate how user-initiated active measurements can be used to both gather new data about LEO internet and assess and compare the performance of individual networks. First, performance metrics that reflect typical internet usage scenarios, such as web browsing and video streaming, are chosen. Next, a test suite is developed to collect data about one's network. It also augments this data with location information to aid in later comparison. This data can then be used for comparisons between individual networks as well as in further research. The last step integrates the suit developed into a web based platform that aims to provide a wide variety of information about Starlink's performance, architecture and allow users to gauge the potential benefits of transitioning from terrestrial to LEO internet could provide them.

1 Introduction

Low Earth Orbit (LEO) internet satellite constellations have become an increasingly popular method of accessing the internet in recent years. They promise fast and reliable internet to people living in remote or underserved regions [42]. This helps solve the digital divide highlighted in the UN-Habitat report on global digital access disparities [45]. LEO satellite internet also provides internet access during air, land, and sea [38, 41, 43, 44] travel, enhancing both passenger comfort as well as aiding in remote rescue operations and business endeavors.

Several LEO internet satellite constellations exist, including Starlink [39], Project Kuiper [2], and OneWeb [15]. Among them, Starlink is currently the largest, with over 5 million registered users [36] as well as more than 7500 active satellites in orbit [24]. Consequently, this research primarily focuses on Starlink, though the methods developed can be easily extended to other constellations.

Despite high interest in the technology, there exists only limited research into its performance due to significant challenges in its measurement and the novelty of LEO internet. For instance, limitations in accurate IP based geolocation capability and the lack of public IPv4 addresses for most Starlink users [47], limit researchers' ability to collect accurate global performance data without deploying complex measurement infrastructure. While some work has examined localized [22, 25] as well as global [26, 47] Starlink performance, there is no research enabling the comparison of any one local network to the local performance of Starlink. This creates not only a knowledge gap, but also a challenge for individuals evaluating the potential benefits of the transition from terrestrial to LEO internet.

This research aims to demonstrate how user-initiated active measurements can be used to address this gap while also solving the challenge individuals face. This goal is achieved by developing a test suite that allows users to assess their own network performance in comparison with other local terrestrial and LEO networks. In addition to providing insights for individual users, it also allows for gathering data that contributes to further research efforts.

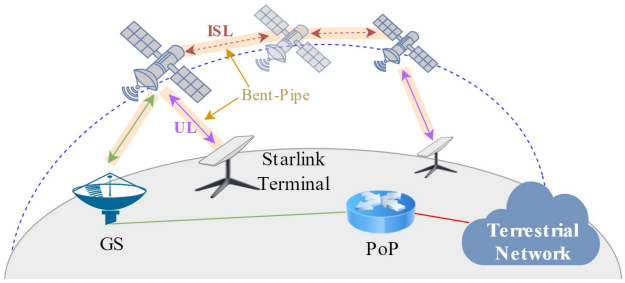

Figure 1: Starlink's architecture diagram. Reproduced from Nitinder Mohan et al., "A Multifaceted Look at Starlink Performance," in Proceedings of the 2024 Web Conference (WWW'24), ACM, 2024. https://doi.org/10.1145/3589334.3645328. © 2024 Copyright held by the owner/author(s). Licensed to ACM.

This suite is then integrated with a larger platform that provides end-users with information on Starlink's future predicted performance as well as its infrastructure. While there exist several tools that allow for assessing Starlink's performance and infrastructure [33, 40, 46] and while network measurement tools like speed test are widely used [7, 20, 32], there is no unified platform combining this information and enabling meaningful cross-network performance comparisons.

2 Background

In the context of internet measurements, telemetry entails collecting data about the performance of a network. Those measurements can be categorized as active when performed by the user or passive when relying on data collected by third parties [27]. This paper will focus on active measurements initiated by an end-user and their benefit when assessing the performance of a network. Multi-layered telemetry entails measuring different network metrics (e.g. latency and throughput) over different protocols, this provides a holistic look into given networks performance instead of focusing on a single use case scenario.

The following description of Starlink's architecture is largely based on the work by Mohan et al. [26]. The architecture of Starlink is unique compared to terrestrial internet. The constellation consists of thousands of satellites orbiting at approximately 550 km above Earth's surface. The majority of the satellites have an orbit with the inclination of 53°but there are also some that use 70°and 97.7°orbits to serve the polar regions. As can be seen in Figure 1, Starlink uses a "Bent-Pipe" architecture. Points of Presence (PoP) usually located at internet exchange points provide an interface between the terrestrial and Starlink network. Each PoP is then connected to one or more Ground Stations (GS). The Ground Stations use the Ka-band link to connect to the satellites. Each end-user requires a Starlink Terminal called "Dishy". This terminal uses phased-array antennas to connect to a satellite using the Ku-band link. Each satellite can connect to multiple Dishys. It can then relay the connection to a GS forming a "bent-pipe" or if the distance between the end-user

Table 1: Metrics chosen to be measured

Category	Metric	Unit
Web Browsing	Latency to Google DNS	ms
Speed Test	Download Speed	Mb/s
Speed Test	Upload Speed	Mb/s
Speed Test	Latency	ms
Speed Test	Packet Loss	%
Dash streaming	Buffer Level	s
Dash streaming	Bitrate	Kb/s
Dash streaming	Resolution	px
Dash streaming	Framerate	FPS
WebRTC	Latency	ms
WebRTC	Audio Packet Loss	%
WebRTC	Audio Jitter	ms
WebRTC	Audio Bitrate	Kb/s
WebRTC	Video Packet Loss	%
WebRTC	Video Jitter	ms
WebRTC	Video Bitrate	Kb/s

and the GS is too great, the newer satellites [35] are capable of laser Inter Satellite Link (ISL), and can relay the signal between multiple satellites before reaching a GS forming an "extended bent-pipe".

3 Methodology

3.1 NetMet

3.1.1 Introduction to NetMet. NetMet [5] is a browser extension that allows the user to run comprehensive network tests using their browser. It includes a web browsing test, a speed test, and a video streaming test. The web browsing test uses browser's built-in Performance API [29] to measure DNS lookup, server connection, TLS negotiation, and first byte time. It achieves this by using Chrome Extension APIs [11] to launch new browser windows with popular regional websites and monitor the loading process of those pages. The speed test measures upload and download throughput, latency, and packet loss. It utilizes the M-lab's NDT test [8, 19, 20]. The video streaming test is based on the DASH protocol [28]. It utilizes the dash.js [9] and the Akamai test servers [1] to measure bitrate, buffer level, framerate, and resolution of a video stream. As this tool fits the goal of performing multi-layered measurements and is open-source it was chosen as the basis of the test suite. However, because one of the goals of the test suite and the larger platform was to provide the end-user with a seamless test experience, the tool had to be adapted to work as a part of a website instead of a browser extension.

3.1.2 Web Test. Adapting the web browsing test presented the most significant challenge. The NetMet implementation of the test used Chrome Extension APIs to open a new browser window and connect to websites inside it. While the connection was being established, it used the resource timing module of the browser's Performance API. Lastly Chrome Extension APIs were again used to send messages back to the main extension's window about the measured performance. However, this proved impossible to replicate inside a webpage without the access to Chrome Extension APIs due to cross-origin resource sharing (CORS) policy. Because of CORS restrictions, when performing a web request, timing properties are inaccessible to the client unless the server explicitly permits it via the 'Timing-Allow-Origin' header [29]. As this would require acquiring permissions from major website providers around the world it was deemed infeasible. A potential method to circumvent this issue would be to only measure to a server under the researcher's control. However, that would introduce a bias in measurements as the server is based in the Netherlands, users running the measurements from further locations would receive worse results. Due to these limitations, a simplified approach was chosen. The test would be based on latency measurements to website provider edge nodes. This would still give a result indicative of web browsing performance while circumventing the issues detailed above. Google's DNS endpoint was chosen as the target of the measurement due to its ubiquity. Ten HTTP GET requests are sent to "https://dns.google" with caching disabled. The time between the sending of the request and the response arriving is then measured to calculate the latency. The test result is calculated as the median of all the latency values. The median was chosen over mean due to its higher resistance to outliers. The choice to use only ten samples was due to its correlation with total test time, on slow networks, the test could take minutes if a high sample size was chosen, hindering the user experience.

3.1.3 Speed Test. This test was adapted in almost its entirety from NetMet. The only major change made was the process of deciding on the test server. Due to the limited accuracy of IP based geolocation when using Starlink [47] the server chosen to perform the test can sometimes not be the closest one resulting in reduced test accuracy. However, there exists a dataset created by another researcher participating in this project [14], that provides a mapping of cities to their best-performing servers locations. This dataset was augmented to map the best-performing server for each region and country, based on how many cities in a region or country use a particular server. The method of choosing the server for the measurement is to first check if a direct mapping exists between a city and a server, if so, this mapping is used, then the fallbacks become the best-performing server for the region, the best-performing server for the country, and lastly the standard selection method used in the NDT test [21].

While this approach helps to mitigate the bad server selection issue, it has some major conceptual and implementation limitations. First it requires the mapping to be periodically updated as the internet infrastructure might change in time. Second, while for cities with a direct mapping the advantage of this method is clear as it will always pick the best server as long as the dataset is up to date, in the fallback criteria where it chooses the server based on the region or country of the city the advantage of this method becomes unclear. To assess the difference in performance between these cases and the standard NDT method would require a large scale global study, which is beyond the scope of this work. This issue could be circumvented by running the test using both methods and selecting the better result, this could also be used as a way to acquire the data on the performance of both methods. However, this solution can degrade user experience as the test can take a long time on slower networks and running it twice augments the issue.

It can also be argued that while the chosen server selection method might not always provide a better server it introduces stability as each city will always be mapped to the same server as long as the data set doesn't change, reducing the variance in the collected data. The last major limitation of this method is an implementation one. As the dataset contains mappings to server locations rather than specific servers, a request to M-Lab's NDT Locate API [18] is required to acquire the server URL. The API doesn't provide the ability to filter servers by city location, only region and country level filtering is possible, it also only returns 4 servers with each request. Because in the current version of the dataset only the city and country of each server are stored, a request is made based on the country of the server. This creates an issue in big countries where more than 4 servers are present, this will frequently lead to the request providing no mapping for the city in question. The current implementation falls back to using the standard NDT server selection method if the issue occurs. The better solution would be to augment the dataset with server regions as it's unlikely a region will contain more than 4 servers.

3.1.4 Streaming test. The DASH streaming test was adapted from NetMet with no major changes. The only minor change was the reduction of test runtime from 30 to 20 seconds. This is due to Starlink performing periodic reconfigurations every 15 seconds that can have significant impact on the performance [26]. This ensures that in each test only one such event can occur. The data is then aggregated using the mean. In this case a mean was chosen in order to capture the impact sudden connection quality changes have on the stream.

3.2 WebRTC

3.2.1 Introduction to WebRTC. WebRTC is a technology that enables real time peer-to-peer communications in the browser [12]. It was chosen as an augmentation to the NetMet test suite as it complements it well by covering a common internet use case not covered by NetMet. Its peer-to-peer nature is also unique and no prior research exists on its interaction with Starlink.

3.2.2 WebRTC test. The WebRTC test establishes a video and audio stream between the user and a server-side WebRTC client. It then uses the built-in RTCStatsReport function of the WebRTC API [31] to gather statistics about the connection. Using this method, data about the bitrate, packet loss, and jitter for both the audio and video stream is collected. Latency data is also recorded, but this functionality is limited to Chromium-based browsers due to API limitations [30]. There also exist some additional limitations in the implementation. Due to privacy considerations the video and audio stream sent to the server is blank. Consequently, only the incoming traffic is measured. The peer-to-peer nature of the connection creates a problem: there must be a way for two clients located behind different NATs to connect. To solve this, STUN servers are used [10]. Google's public STUN servers were used in the implementation. STUN however is not perfectly reliable and there might be a case where a connection can't be made. While this could be solved by using TURN servers, this increases complexity and the lack of STUN connectivity can itself be a viable result to the user, due to this TURN servers were not employed. This

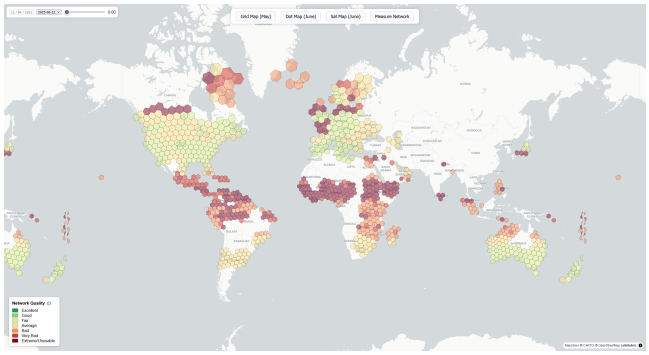

Figure 2: The front page of the web platform.

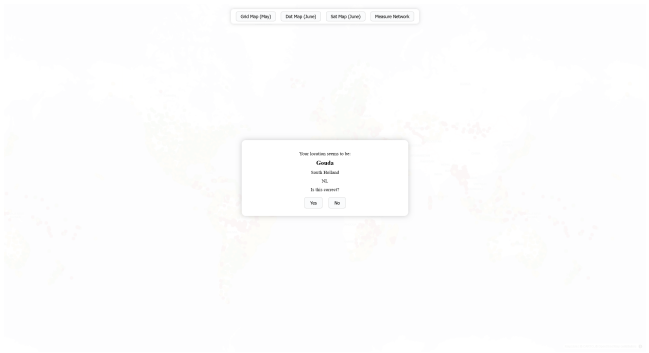

Figure 3: The popup confirming the accuracy of IP geolocation.

introduces bias as the other client is always located on a server in the Netherlands, this leads to reduced accuracy of measurements performed outside of Europe. Overcoming this issue would require the creation of global WebRTC test network which is beyond the scope of this work. There exist however a mitigating factor, from the end-user perspective the comparisons are always performed between nearby networks, this causes the bias on both of them to be similar and cancel out, so while the absolute values are biased, the relative ones are not. Because of this and the uniqueness of this kind of measurement when it comes to Starlink, the test was chosen to be included. The test again takes 20 seconds and uses mean for aggregation for the reasons highlighted at the end of section 3.1.4.

4 Integration and Results

4.1 The integrated platform

The test suite was then integrated into a web based platform providing information about Starlink's performance, it's front page can be seen in Figure 2. The platform was created using the Vue Framework [49] for the front-end and a FastApi [34] back-end, the database was created using PostgreSQL [16]. The database used for the platform is first created using the Global Telemetry Tool created as a different part of this project [13]. The tool can then be used to populate the database with unified Cloudflare and NDT test dataset. Then migrations created using Alembic [4] are run,

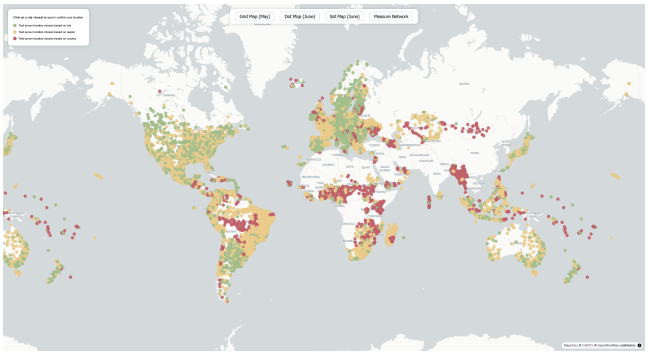

Figure 4: A world map with cities available for selection. Colored based on the NDT server mapping accuracy.

Figure 5: The comparison screen. Keep in mind that the data was created for demonstration purposes and might not be representative of real network performance.

to create the additional tables required for conducting the measurements. This particular tech stack was chosen as it allows for creating complex web applications as well as due to personal preference and previous experience of the members of the research team. The platform's source code is publicly available on GitHub [3]. The further sub-sections will provide insights into the workings and implementation of each of the major components enabling the network performance comparison. Each sub-section will contain a reference to a section in the appendix containing screenshots of the platform showcasing the relevant feature.

4.2 Location gathering

The first step needed for providing the comparison is acquiring the location of the user. The location is collected down to the city level, but some large cities have finer distinctions (e.g. Manhattan and Brooklyn are treated as separate cities). This provides enough accuracy for the comparison while also providing only limited information about end-users precise address. Due to the unreliability of IP based geolocation when using Starlink [47], a different means of precisely acquiring the user location had to be developed. To this end a hybrid approach was chosen, first there is a call to a geolocation API [17], this API was chosen because its free version allows HTTPS requests, has precision down to the city level, provides ASN information used to distinguish between different internet providers, and allows for 50,000 requests per month. After the API returns a response the user gets asked if the inferred location is correct, see Figure 3, if the user confirms the selection the application proceeds to the measurement.

If the location can not be inferred from users IP or the result is not correct the user is given an option to choose their location on a map, see Figure 4. Cities are represented by different colored dots indicating the accuracy of the NDT server mapping discussed in section 3.1.3. The user can then get details about each dot by hovering over it and select it by clicking to proceed to measurement. The map was implemented using MapLibre GL JS library [23]. The city location data uses a free dataset created by Simple Maps [37]. The city data is filtered to only include countries with Starlink operational, converted to the GeoJSON format, and added as a layer to the map.

While this approach provides a good basis for accurate and user-friendly location gathering, it faces some issues. The first and biggest issue is the lack of standardization in location names. Countries are referred to by standardized two-letter codes [48] and the dataset contains both special character containing and ascii variant of city names, regions however lack this standardization, the free version of the dataset includes only localized region names (e.g. Zuid-Holland), on the other hand the geolocation API uses english region names. This causes comparisons made by region to be unreliable, an effort was taken in the implementation to mitigate this issue as much as possible but it is not a perfect solution. The better solution would be to use a paid dataset, either the premium version of the Simple Maps dataset or the dataset provided by the author of the geolocation API. This would also help solve an issue of data sparsity. The free dataset only provides data for selected larger cities. This causes regions like Alaska to have only a few cities to select from. To meet the goal of allowing users in remote places access to accurate measurements, upgrading the dataset should be a priority before launching the service publicly.

4.3 Measurement

After user location is confirmed the sequence of test from the suit are performed. They are launched sequentially as to not impact each other. The implementation details for this step are provided in section 3. The speed and streaming tests use an interface adapted from NetMet while running. The WebRTC test has an interface based on the streaming test due to their similarity. This interface is visible while the tests are running to give the user proper feedback about their progress.

4.4 Comparison

The comparison for all tests is performed in the same manner. When a test finishes, it sends its result along with the user location to the server. The server then queries the database to find tests performed in the same city. However, if less than ten test results are available it then tries to query the data based on the region of the user. If there are again less than ten tests it queries the data on a country level, as this has greatly reduced accuracy, the query only returns archived Starlink results, which due to it's global nature and unified architecture has less variation over large distances than terrestrial,

where some regions might be served by high speed fiber internet and others by radio internet. The fallback threshold was chosen arbitrarily and should be adjusted based on the dataset size and user feedback. The query returns top hundred results sorted by time, Starlink results are provided with a bias that makes them always appear on top of the list, as long as the measurement was taken in the last month. A threshold of 100 was chosen to balance the sample size, data clarity, and response size. It should be further tweaked based on user testing.

This data is then sent back to the user and presented in the form of scatter plots, see Figure 5. The plots were created using the Chart.js library [6], it provides a large amount of customization options that make it easy to tweak existing and add new charts in the future. The user can then hover on the highlighted dot representing their measurement to see its details. For speed, streaming, and rtc tests multiple charts corresponding to the different metrics can be viewed. This remains largely a proof of concept, many different visualization styles for this data could be created, that will allow users to find one that best fits their preferences.

4.5 Cold start

While the speed test data can be compared to unified NDT and Cloudflare dataset created by another researcher participating in this project [14], all the other tests suffer from a cold start. There are no publicly available datasets containing this data and so the only comparisons that can be made is with the data collected by users using the platform. This might slow down the adoption of the site due to limited functionality at the launch. However, this problem is somewhat counteracted by the wide scope of services provided by the platform, the additional services could help build a user base even when the active measurement suite has limited functionality.

5 Conclusion and Future work

User-initiated active measurements are a powerful tool for evaluating Starlink and other network performance. They create unique opportunities for both researchers and users to learn more about networks. While passive data enables large-scale system comparisons, it cannot be used to directly compare individual networks, a gap filled by active measurements. Active measurements can also be used as a source of passive data, making global data collection critical for future research on terrestrial and LEO networks. The test suite and platform created during this research enable the collection of varied network performance data while providing users with a powerful tool for gaining deeper insight about Starlink's performance.

This research also showed some limitations active measurements face when run inside a website environment. The most important insight is that security policies like CORS limit one's ability to gather meaningful data about the performance of third-party websites. This shows that for accurate measurements of such data, tools like browser extensions have a clear advantage. This reduces the ability to gather crowd-sourced data on those metrics, as the need to install a browser extension may reduce user participation.

On the other hand, tests run from inside a website face fewer limitations while measuring the performance of tasks like video streaming or conferencing. Due to this and the limited work done on measuring the interaction of those technologies with Starlink, this suggests a clear direction for further research, to provide new insights into LEO internet. Data collected by the platform developed during this research can then be used to aid in this further work.

Some work still remains that should be addressed before the platform can be publicly launched. Most notably extensive testing has to be performed on a wide variety of systems to ensure a seamless experience. There are also some considerations to be made with the third party tools and datasets used, a lot of them were chosen due to being freely available, for the production version, choosing more capable paid tools, should be considered to reduce some of the limitations encountered during development.

6 Responsible research

While no sensitive user data was collected during this research, the platform developed as part of it will store and handle such information. A clear privacy policy was provided, which users can access and agree to before any data is collected. An active effort was made to minimize the risk of identifying users from the data collected. While network performance data is not considered sensitive, location, time, and internet provider (ISP) data could be. Because of this, the location is collected only at a city level, as it provides sufficient accuracy without exposing any particular addresses. ISP data, even when combined with city and time information, exposes no additional details about the user, as the list of ISPs operating in each region is already publicly available. Considering all this, it is unlikely that any additional sensitive user data can be inferred from the collected information.

All the code and data used in the development, as well as the final product were made publicly available on GitHub [3]. This allows other researchers to verify the validity of the methods used, and provides a way to replicate the results and conclusions reached by this research.

An effort was made to minimize the appearance of any biases in the data collected. To achieve this, the use of centralized servers was largely avoided, as this could introduce extensive bias, considering the global scope of the work. However, it could not be eliminated completely. The WebRTC test had to use centralized servers due to technical limitations. To minimize its impact, the existence of this limitation was documented so any individuals using the data gathered are aware of its existence.

The references used as the basis of this work were based on known reputable sources such as ACM. When information could not be sourced from scientific work published in reputable journals, it was checked to appear multiple times in different articles, websites, blogs, and other similar public sources. All sources of information, data, and code were credited and license terms were followed.

References

[1] Akamai. 2025. *Akamai Stream Validation and Player Test.* Retrieved 2025-05-23 from https://players.akamai.com/

[2] Amazon. 2025. *Projec Kuiper.* Retrieved 2025-06-20 from https://www.aboutamazon.com/what-we-do/devices-services/project-kuiper

[3] Christiaan Baraya, Cristian Benghe, Janusz Urbański, and Vlad-Stefan Graure. 2025. *Global Telemetry Data Processing System.* Retrieved 2025-06-22 from https://github.com/TUD-BScResearchProject-6079/global-telemetry-cli

[4] Michael Bayer. 2025. *Alembic.* Retrieved 2025-06-20 from https://github.com/sqlalchemy/alembic

[5] Rohan Bose. 2025. *NetMet: A Tool for Network Measurement.* Retrieved 2025-06-20 from https://github.com/boserohan/netmet

[6] Chart.js. 2025. *Simple yet flexible JavaScript charting library for the modern web.* Retrieved 2025-06-20 from https://www.chartjs.org/

[7] Cloudflare. 2025. *Internet Speed Test - Measure Network Performance | Cloudflare.* Retrieved 2025-06-10 from https://speed.cloudflare.com/

[8] Code for Science & Society (CS&S). 2025. *Measurement Lab.* Retrieved 2025-05-23 from https://www.measurementlab.net/

[9] Dash Industry Forum. 2025. *dash.js.* https://github.com/Dash-Industry-Forum/dash.js

[10] Getstream. 2025. *Learn STUN & TURN Servers on WebRTC.* Retrieved 2025-06-20 from https://getstream.io/resources/projects/webrtc/advanced/stun-turn/

[11] Google. 2025. *API reference.* Retrieved 2025-06-20 from https://developer.chrome.com/docs/extensions/reference/api

[12] Google. 2025. *WebRTC.* Retrieved 2025-06-20 from https://webrtc.org/

[13] Vlad-Stefan Graure. 2025. *Global Telemetry Data Processing System.* Retrieved 2025-06-20 from https://github.com/TUD-BScResearchProject-6079/global-telemetry-cli

[14] Vlad-Stefan Graure. 2025. Multi-Layered Telemetry Assessing Global Performance of LEO Internet Providers: Towards a Global Telemetry System for Evaluating LEO ISP Performance.

[15] Eutelset Group. 2025. *OneWeb.* Retrieved 2025-06-20 from https://oneweb.net/

[16] PostgreSQL Global Development Group. 2025. *PostgreSQL: The World's Most Advanced Open Source Relational Database.* Retrieved 2025-06-10 from https://www.postgresql.org/

[17] IpInfo. 2025. *IpInfo.* Retrieved 2025-06-10 from https://ipinfo.io/

[18] Measurement Lab. 2025. *Locate API v2.* Retrieved 2025-06-20 from https://www.measurementlab.net/develop/locate-v2/

[19] Measurement Lab. 2025. *ndt7-js.* https://www.speedtest.net/

[20] Measurement Lab. 2025. *Speed Test by Measurement Lab.* Retrieved 2025-06-10 from https://speed.measurementlab.net/

[21] Phillipa Gill Loqman Salamatian. 2025. *How M-Lab Determines User Location and Selects Servers.* Retrieved 2025-06-20 from https://www.measurementlab.net/blog/improving-m-lab-geolocation/

[22] Sami Ma, Yi Ching Chou, Haoyuan Zhao, Long Chen, Xiaoqiang Ma, and Jiangchuan Liu. 2022. Network Characteristics of LEO Satellite Constellations: A Starlink-Based Measurement from End Users. arXiv:2212.13697 [cs.NI] https://arxiv.org/abs/2212.13697

[23] MapLibre. 2025. *MapLibre GL JS.* Retrieved 2025-06-20 from https://maplibre.org/maplibre-gl-js/docs/

[24] Jonathan McDowell. 2025. *Satellite statistics: Satellite and Debris Population.* Retrieved 2025-06-20 from https://planet4589.org/space/stats/active.html

[25] François Michel, Martino Trevisan, Danilo Giordano, and Olivier Bonaventure. 2022. A first look at starlink performance. In *Proceedings of the 22nd ACM Internet Measurement Conference* (Nice, France) *(IMC '22)*. Association for Computing Machinery, New York, NY, USA, 130–136. https://doi.org/10.1145/3517745.3561416

[26] Nitinder Mohan, Andrew E. Ferguson, Hendrik Cech, Rohan Bose, Prakita Rayyan Renatin, Mahesh K. Marina, and Jörg Ott. 2024. A Multifaceted Look at Starlink Performance. In *Proceedings of the ACM Web Conference 2024* (Singapore, Singapore) *(WWW '24)*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3589334.3645328

[27] Venkat Mohan, YR Janardhan Reddy, and K Kalpana. 2011. Active and passive network measurements: a survey. *International Journal of Computer Science and Information Technologies* 2, 4 (2011), 1372–1385.

[28] Mozilla. 2025. *DASH Adaptive Streaming for HTML video.* Retrieved 2025-06-20 from https://developer.mozilla.org/en-US/docs/Web/API/Media_Source_Extensions_API/DASH_Adaptive_Streaming

[29] Mozilla. 2025. *Resource timing.* Retrieved 2025-05-23 from https://developer.mozilla.org/en-US/docs/Web/API/Performance_API/Resource_timing

[30] Mozilla. 2025. *RTCIceCandidatePairStats: currentRoundTripTime property.* Retrieved 2025-06-20 from https://developer.mozilla.org/en-US/docs/Web/API/RTCIceCandidatePairStats/currentRoundTripTime

[31] Mozilla. 2025. *RTCStatsReport.* Retrieved 2025-06-20 from https://developer.mozilla.org/en-US/docs/Web/API/RTCStatsReport

[32] Ookla. 2025. *Speedtest by Ookla - The Global Broadband Speed Test.* Retrieved 2025-06-10 from https://www.speedtest.net/

[33] Mike Puchol. 2025. *Starlink Satellite Coverage Visualization.* Retrieved 2025-04-22 from https://starlink.sx/

[34] Sebastián Ramírez. 2025. *FastAPI.* Retrieved 2025-06-10 from https://fastapi.tiangolo.com/

[35] Via Satellite. 2021. *Latest Starlink Satellites Equipped with Laser Communications, Musk Confirms.* Retrieved 2025-06-20 from https://www.satellitetoday.com/launch/2021/01/25/latest-starlink-satellites-equipped-with-laser-communications-musk-confirms/

[36] SatNews. 2024. *Analyst: Starlink at 5m users.* Retrieved 2025-05-02 from https://news.satnews.com/2024/12/19/analyst-starlink-at-5m-users/

[37] SimpleMaps. 2025. *World Cities Database Basic.* Retrieved 2025-06-20 from https://simplemaps.com/data/world-cities

[38] SpaceX. 2025. *ROAM WITH STARLINK.* Retrieved 2025-05-02 from https://www.starlink.com/roam

[39] SpaceX. 2025. *Starlink.* Retrieved 2025-05-02 from https://www.starlink.com/

[40] SpaceX. 2025. *Starlink Coverage Map.* Retrieved 2025-04-22 from https://www.starlink.com/map

[41] SpaceX. 2025. *Starlink For Aviation.* Retrieved 2025-05-02 from https://www.starlink.com/business/aviation

[42] SpaceX. 2025. *Starlink For Homes.* Retrieved 2025-05-02 from https://www.starlink.com/residential

[43] SpaceX. 2025. *Starlink For Land Mobility.* Retrieved 2025-05-02 from https://www.starlink.com/business/mobility

[44] SpaceX. 2025. *Starlink For Maritime.* Retrieved 2025-05-02 from https://www.starlink.com/business/maritime

[45] UN-Habitat. 2023. *Assessing the Digital Divide.* Retrieved 2025-05-02 from https://unhabitat.org/programme/legacy/people-centered-smart-cities/assessing-the-digital-divide

[46] Unknown. 2025. *Unofficial Starlink Global Gateways & PoPs.* Retrieved 2025-05-02 from https://www.google.com/maps/d/u/0/viewer?mid=1805q6rlePY4WZd8QMOaNe2BqAgFkYBY&hl=en_US&ll=-3.81666561775622e-14,0&z=2

[47] Bingsen Wang, Xiaohui Zhang, Shuai Wang, Li Chen, Jinwei Zhao, Jianping Pan, Dan Li, and Yong Jiang. 2024. A Large-Scale IPv6-Based Measurement of the Starlink Network. arXiv:2412.18243 [cs.NI] https://arxiv.org/abs/2412.18243

[48] Wikipedia. 2025. *List of ISO 3166 country codes.* Retrieved 2025-06-20 from https://en.wikipedia.org/wiki/List_of_ISO_3166_country_codes

[49] Evan You. 2025. *Vue.js.* Retrieved 2025-06-10 from https://vuejs.org/