Quantifying TOR and VPNs Performance

Anish Budhi
Rutgers University
Piscataway, NJ, USA
akb147@scarletmail.rutgers.edu

Daniel Ojeda
Rutgers University
Piscataway, NJ, USA
deo50@scarletmail.rutgers.edu

Nilang Patel
Rutgers University
Piscataway, NJ, USA
ngp50@rutgers.edu

Liam O'Neill
Rutgers University
Piscataway, NJ, USA
ljo38@rutgers.edu

I. Abstract

The Onion Router (TOR) Browser and Virtual Private Networks (VPNs) are widely recognized as privacy and security-enhancing tools for web browsing. However, these tools come with performance trade-offs that have detered broader adoption. In this project, we aim to quantify the performance differences between TOR, VPNs, and unaltered web browsers by analyzing four key metrics, page load times, download speeds, network latency, and first-byte times. By conducting tests for these metrics over different websites meant to represent the average person's internet usage, we seek to provide insights into the usability and practicality of these privacy tools under everyday circumstances. We hope the results of this analysis will contribute to our understanding of the challenges and limitations of adopting TOR and VPNs for everyday web browsing.

II. Introduction

In this project, we focus on evaluating the performance of three web browsing configurations:

1) Browsing through TOR, a system designed for anonymity and privacy.
2) Browsing through VPNs, which provide encrypted communication and conceal the user's IP address.
3) Browsing without any additional privacy tools, using standard browsers like Chrome.

We tested over these 3 configurations on a variety of websites categorized into static content (e.g., documentation sites like Python.org), dynamic content (e.g. Amazon), and multimedia-rich sites (e.g., YouTube and Netflix).

To assess performance, we measured:

1) **Page Load Times:** This metric reflects the user's experience when loading web pages. It is critical to evaluate the impact on browsing speed.
2) **Download Speeds:** As a fundamental indicator of throughput, this metric shows how efficiently data is transferred.
3) **Network Latency:** This measures the delay in transmitting data packets and is crucial for interactive applications and real-time communication.
4) **First Byte Time:** This measures the time taken for the first byte of data to reach the browser from the server. It provides insights into the responsiveness of the connection.

III. Methodology

A. *Page Load Times:*

To measure the page load time for websites across different browsing configurations, we conducted experiments using Chrome, Chrome with Proton VPN (connected to US-FREE17), and the Tor Browser. For the Chrome and Proton VPN tests, we utilized Python's Selenium library along with ChromeDriver to automate browser navigation and measure page load times. The script executed a series of tasks, including navigating to a predefined list of websites, calculating the time difference between the browser's navigation start and the load event end, and averaging these measurements over 100 cycles for each site. The tests were run in headless mode, with GPU acceleration disabled and WebGL fallback enabled.

For the Tor Browser tests, Selenium was configured with GeckoDriver and connected to the Tor proxy on 127.0.0.1:9150. Firefox's manual proxy settings ensured all traffic was routed through Tor. The Tor Browser's SOCKS5 proxy setup was used to anonymize requests. The page load time was measured by recording the elapsed time between initiating a page request and its completion. Similar to the Chrome-based tests, this process was repeated for 100 cycles per site, with the first cycle excluded to account for warm-up effects.

All tests were conducted under equal network conditions, around the same time, with the same internet connection. These conditions were not entirely controlled, but the equality in conditions helped to minimize variability in results. Bandwidth usage during the tests was monitored using the psutil library. Network I/O statistics were sampled every second during the test runs, providing metrics on average and maximum bandwidth usage. Results were written to output files using a custom logging class to facilitate real-time monitoring and post-test analysis.

B. *Download Speeds:*

The measurements for download speeds were collected using Python's Requests library, first using the Tor proxy, then without, and last with Proton VPN enabled through servers in three different locations. Specifically the servers used for measuring download speeds were US-FREE34, NL-FREE1, and JP-FREE28 located in the United States, the Netherlands, and Japan respectively. The python script was able to measure download speed by requesting a page, tracking the time taken to download the data, and then dividing the size of the data by that amount of time. The pages to request from were predefined, and 100 trials were done for each to accurately calculate the average download speed.

All download speed tests were done on the same network, at approximately the same time, with no noticeable changes in network conditions occurring during the testing. For these tests, the results were not written to an output file, but rather recorded in a spreadsheet, the contents of which are shown in TABLE 2 in the results section.

C. Network Latency:

To measure the latency for different browsing configurations, we conducted a series of tests using Chrome (No VPN), Chrome with Proton VPN, and Tor Browser. For each configuration, a predefined list of websites, including lightweight and media-rich platforms, was tested to analyze performance variations. Latency was calculated as the time between sending a request and receiving the response.

The tests for Chrome (No VPN) and Chrome with VPN were automated using Python as well as Javascript to confirm the results, to make sure I am getting consistent results. Proton VPN was configured with multiple server locations, adding an encryption layer and routing traffic through secure servers. For Tor Browser, it was connected to Tor's SOCKS5 proxy, ensuring all requests were anonymized through Torâs multi-relay network. Tor was also configured to disable all forms of caching to maintain consistency across configurations.

Each website was tested 100 times under all configurations, and the results were averaged to calculate reliable latency values. The tests were conducted using the same internet connection and at the same time to minimize external variability. Results were logged and visualized in bar charts, highlighting the differences in latency. This approach provided insights into how routing, encryption, and privacy mechanisms influence performance, demonstrating the trade-offs between speed and anonymity in different browsing setups.

D. First Byte Time:

To measure the Time To First Byte (TTFB) for various websites under different browsing configurations, we conducted experiments using Chrome, Chrome with Proton VPN (connected to US-FL211, AR-17, IT-64, AE-9, JP-59, ZA-47, NZ-14), and the Tor Browser. For the Chrome-based tests, we used Python's Selenium library in conjunction with ChromeDriver (Browser Version: 131.0.6778.109 and ChromeDriver Version: 131.0.6778.87) to automate browser interactions. The script was designed to navigate through a predefined list of websites and measure TTFB by calculating the difference between the browser's 'requestStart' and 'responseStart' performance metrics and average these measurements over 100 runs for each site. The Chrome browser was configured with multiple preferences to disable caching mechanisms, ensuring consistent running conditions. Additionally, the browser was run in incognito mode to minimize residual data effects.

For the Tor Browser tests, Selenium was configured with GeckoDriver (Tor Browser Version: 128.5.0 and Geckodriver Version: 0.35.0) and connected to Tor's SOCKS5 proxy at 127.0.0.1:9150. This setup ensured that all traffic was routed through the Tor network, providing anonymity for each request. Similar to the Chrome tests, the script navigated to the same list of websites, measured the TTFB by capturing the relevant performance metrics, and averaged the results over 100 runs per site. While Tor inherently manages privacy and caching, Tor Browser was configured to disable disk, memory, and offline caches to eliminate the slightest possibility of cached elements influencing the results and address any potential concerns about caching affecting the measurements.

While absolute control over network variables was challenging, the tests were performed at the same time after midnight using the same internet connection to minimize variability. The results of these experiments were logged to both the console and a file, ensuring that all TTFB measurements were captured accurately and were accessible for further analysis.

IV. Results

A. Page Load Times:

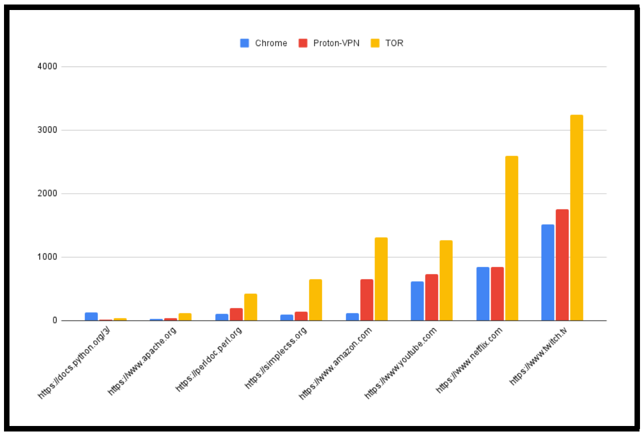


Fig. 1. Page Load Times by Browsing Configuration by Website (ms)

Browsing Configuration	Chrome	Proton-VPN	TOR
https://simplecss.org	92.67 ms	142.41 ms	649.99 ms
https://docs.python.org/3/	126.82 ms	13.66 ms	38.23 ms
https://www.apache.org	26.12 ms	36.63 ms	122.87 ms
https://perldoc.perl.org	105.41 ms	195.45 ms	426.81 ms
https://www.amazon.com	117.87 ms	646.57 ms	1312.67 ms
https://www.airbnb.com	682.53 ms	907.18 ms	972.23 ms
https://www.tripadvisor.com	211.41 ms	285.75 ms	665.26 ms
https://www.youtube.com	619.36 ms	728.37 ms	1265.05 ms
https://www.tiktok.com	385.16 ms	438.57 ms	1398.38 ms
https://www.netflix.com	850.46 ms	844.34 ms	2599.40 ms
https://www.twitch.tv	1519.01 ms	1750.83 ms	3240.75 ms

TABLE I
Page Load Times by Browsing Configuration

The measured page load times across the three browsing configurations showed significant differences. Chrome consistently had the fastest load times, with lightweight websites like https://simplecss.org and https://www.apache.org loading in under 100 ms. These sites require minimal resources, enabling

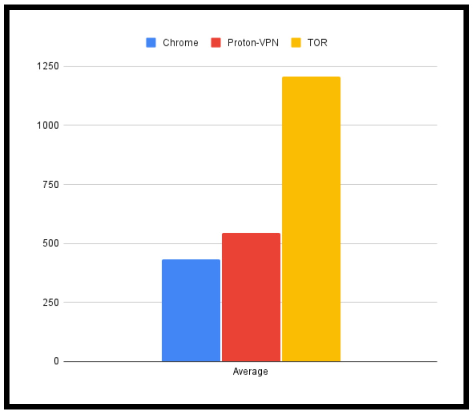


Fig. 2. Average Page Load Times by Browsing Configuration (ms)

Chrome's optimized performance. In contrast, Tor's onion-routing process increased load times, with even the simple website taking over 600 ms to load.

Mid-level websites such as https://www.youtube.com and https://www.amazon.com showed larger performance gaps. Amazon loaded in 117.87 ms on Chrome but exceeded 1300 ms on Tor. Tor's performance for both websites was similar, but Chrome was much faster for Amazon than YouTube. This may be due to YouTube's reliance on QUIC, while Amazon uses both TCP and QUIC with TCP fallback, but further testing would be useful to confirm if these results are based on protocol effects.

Dynamic, resource-heavy websites like https://www.netflix.com and https://www.twitch.tv also exhibited large performance gaps. Netflix took 850.46 ms on Chrome but exceeded 2500 ms on Tor, while Twitch spiked to over 3200 ms on Tor compared to 1519.01 ms on Chrome. These sites demand significant resources due to their high reliance on video streaming and real-time content delivery. Tor's onion-routing mechanism seems to have amplified the delays of resource-heavy pages.

Anomalies in the results include Chrome's unusually high load time for https://docs.python.org/3/. This likely occurred because Chrome refreshed the page so quickly, over 100 times per second, that the program failed to measure properly, resulting in a bug. Some websites also could not be tested due to limitations, like https://www.bestbuy.com, which was excluded because buggy elements caused load times to exceed 10x those of other sites and triggered runtime errors, threatening program stability. Additionally, data for Tor on certain websites, highlighted in red in *TABLE I*, had to be excluded because the pages redirected to bot-check and CAPTCHA screens instead of home pages, making comparisons invalid.

On average, for our representative valid website set, page load times were 432.22 ms for Chrome, 544.78 ms for Proton VPN, and 1206.97 ms for Tor. This aligns with the bandwidth measurements during testing, where Chrome achieved the highest average bandwidth at 3.87 MB/s, followed by Proton VPN at 2.91 MB/s, and Tor at a significantly lower 110.98 KB/s. The maximum bandwidth measurements, 6.82 MB/s for Chrome, 5.56 MB/s for Proton VPN, and 710.92 KB/s for Tor, highlight how constraining Tor's onion-routing mechanism is, even when compared to a free VPN. However, while Tor's bandwidth was over 30 times slower than Chrome, its page load times were not proportionally slower, which could be attributed to factors such as caching or reduced content prioritization for lower-quality connections.

B. Download Speeds:

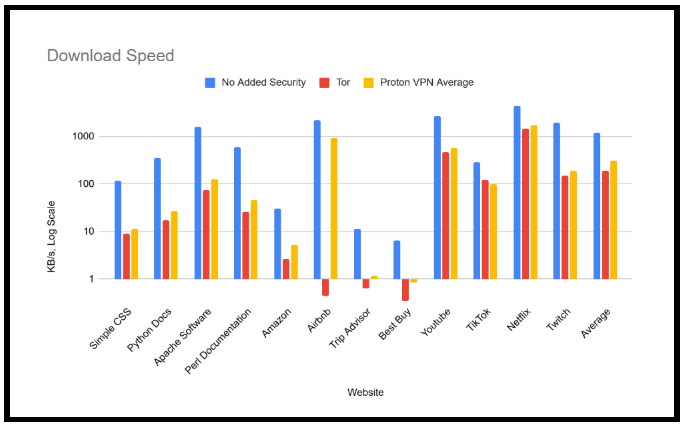


Fig. 3. Download Speed by Configuration by Website (KB/s, Log Scale)

Configuration	No Added Security	TOR	Proton-VPN (US / NL / JP)
https://simplecss.org	117.58	9.02	16.05 / 12.32 / 6.23
https://docs.python.org/3/	361.68	17.06	40.61 / 27.11 / 14.68
https://www.apache.org	1593.95	75.73	181.88 / 130.04 / 69.41
https://perldoc.perl.org	594.4	26.02	68.18 / 47.35 / 24.41
https://www.amazon.com	30.12	2.65	9.64 / 4.42 / 1.9
https://www.airbnb.com	2196.06	0.43	1134.85 / 972.32 / 680.22
https://www.tripadvisor.com	11.39	0.63	1.67 / 1.25 / 0.64
https://www.bestbuy.com	6.51	0.34	1.16 / 0.9 / 0.48
https://www.youtube.com	2783.21	464.78	764.4 / 625.47 / 336.51
https://www.tiktok.com	288.86	120.55	162.01 / 141.85 / 1.19
https://www.netflix.com	4480.83	1459.18	2550.83 / 1712.18 / 943.69
https://www.twitch.tv	1956.4	151.49	270.06 / 196.88 / 105.97

TABLE II
DOWNLOAD SPEED BY CONFIGURATION AND VPN LOCATION (KB/S)

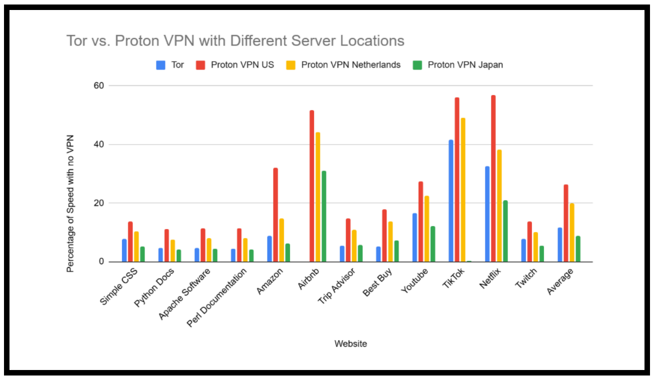


Fig. 4. Download Speed Comparison Between Tor and Proton-VPN Locations

While the actual values of download speeds varied greatly with the sizes of the individual pages, an analysis of the differences between measurements shows that using Tor, a VPN, or neither also drastically changes download speeds.

Using neither was consistently the fastest, with Tor and VPNs both reducing download speeds when enabled. To make this visible despite the large variations cause by page size, Fig. 3 uses a log scale to show the recorded download speed, with VPN measurements for each location averaged. It also shows that, for all websites tested but one (https://www.tiktok.com/), Proton VPN is faster than using Tor. This is also reflected in the average, which has Proton VPN being about 60% faster than Tor.

TABLE II shows all of the data collected, including for each different server location with Proton VPN. We again see here that using neither Tor nor a VPN is by far the fastest, with most download speeds through either one being less than half of the original speed. This table also shows where there were a few outliers in the data. First, when using Tor to connect to https://www.airbnb.com/ the download speed is less than 0.1% of the speed without or when using a VPN. The limitations of this project mean that the exact cause cannot be diagnosed, but it is likely due to https://www.airbnb.com/ redirecting our connection request to a much simple page than their standard home page, which could be caused by a bot detection system they use, the location of the Tor exit node during these tests, or a combination of both. The other noticeable outlier occurred when testing https://www.tiktok.com/ through a Proton VPN server located in Japan. This gave a download speed of less than 1% of what was measured from other locations, indicating again that some sort of redirection to a simpler page occurred, this time being specifically caused by the VPN server location. Overall, neither of these outliers caused a qualitative difference in the results, as the average speeds remain similar if these values are removed.

Fig. 4 shows the download speeds of Tor and Proton VPN with each tested location as a percentage of the speed measured when neither is used. This is done to provide an alternate visualization of the data that again accounts for the variation caused by website size, this time also showing the differences between VPN server locations. It is clear here that VPN servers in closer locations provide significantly faster speeds. Additionally, Tor's speeds are only sometimes better than the slowest VPN location tested, with the fastest VPN location yielding download speeds that are almost always twice as fast as Tor or faster. We can also see these same trends in the averages across websites, with the Japan server being similar to Tor, and the US server being more than twice as fast as Tor. Overall, the data strongly indicate a major advantage to using a VPN for security rather than Tor in regards to download speeds.

C. Network Latency:

The graph compares the latency across three configurations: Chrome (No VPN), Chrome with VPN, and Tor Browser, for various websites. Chrome (No VPN) consistently demonstrates the lowest latency, typically under 100ms for lightweight websites like 'simplecss.org' and 'docs.python.org' and slightly higher for media-heavy platforms like 'netflix.com'. Chrome with VPN exhibits moderate latency, ranging from 200ms to

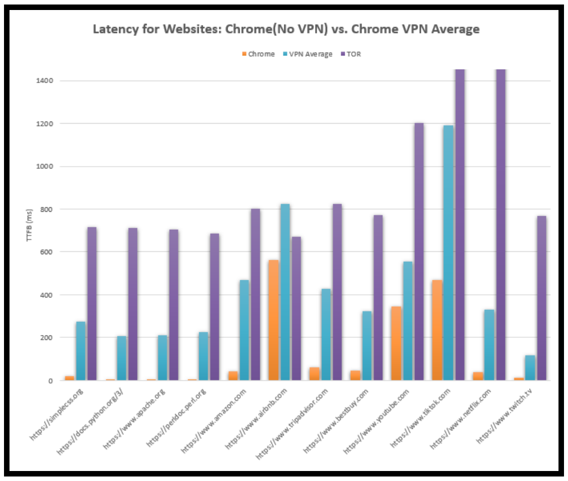
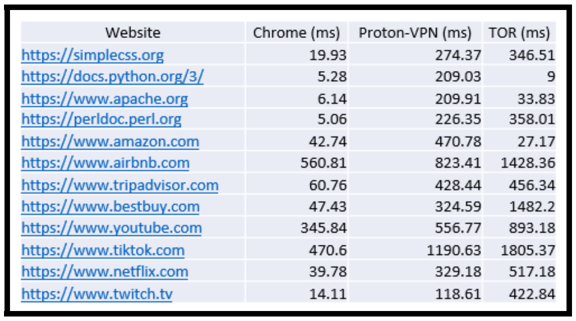


Fig. 5. Latency by Browsing Configuration (ms)



Website	Chrome (ms)	Proton-VPN (ms)	TOR (ms)
https://simplecss.org	19.93	274.37	346.51
https://docs.python.org/3/	5.28	209.03	9
https://www.apache.org	6.14	209.91	33.83
https://perldoc.perl.org	5.06	226.35	358.01
https://www.amazon.com	42.74	470.78	27.17
https://www.airbnb.com	560.81	823.41	1428.36
https://www.tripadvisor.com	60.76	428.44	456.34
https://www.bestbuy.com	47.43	324.59	1482.2
https://www.youtube.com	345.84	556.77	893.18
https://www.tiktok.com	470.6	1190.63	1805.37
https://www.netflix.com	39.78	329.18	517.18
https://www.twitch.tv	14.11	118.61	422.84

Fig. 6. Data Table for Latency

600ms, with the additional overhead due to encryption and routing through VPN servers. In contrast, the Tor Browser shows the highest latency, often exceeding 800ms and reaching over 1400ms for websites like 'netflix.com', due to its multi-relay routing and layered encryption. The chart highlights the trade-off between speed and privacy, with Chrome offering the fastest performance, VPN balancing speed and security, and Tor prioritizing anonymity at the cost of significantly increased latency.

D. First Byte Time:

Fig. 7 shows the Time to First Byte (TTFB) for different websites tested across various VPN server locations. The red line represents the global mean, showing the average performance of all servers for each website. Fig. 7 and Table III show that static websites such as SimpleCSS, Python.org, Apache, and Perldoc have the fastest TTFB across all regions, with global means of 78.99 ms, 33.66 ms, 40.33 ms, and 78.51 ms, respectively. This is likely because static content requires minimal server-side processing. Additionally, these websites showed slight variations in TTFB across regions, indicating minimal geographical impact on performance.

Dynamic content websites, such as Amazon, Airbnb, and Tripadvisor, exhibited higher TTFB values due to the additional complexity of server-side processing. Airbnb's TTFB ranged from 90.21 ms in Miami to 286.73 ms in Johannesburg,

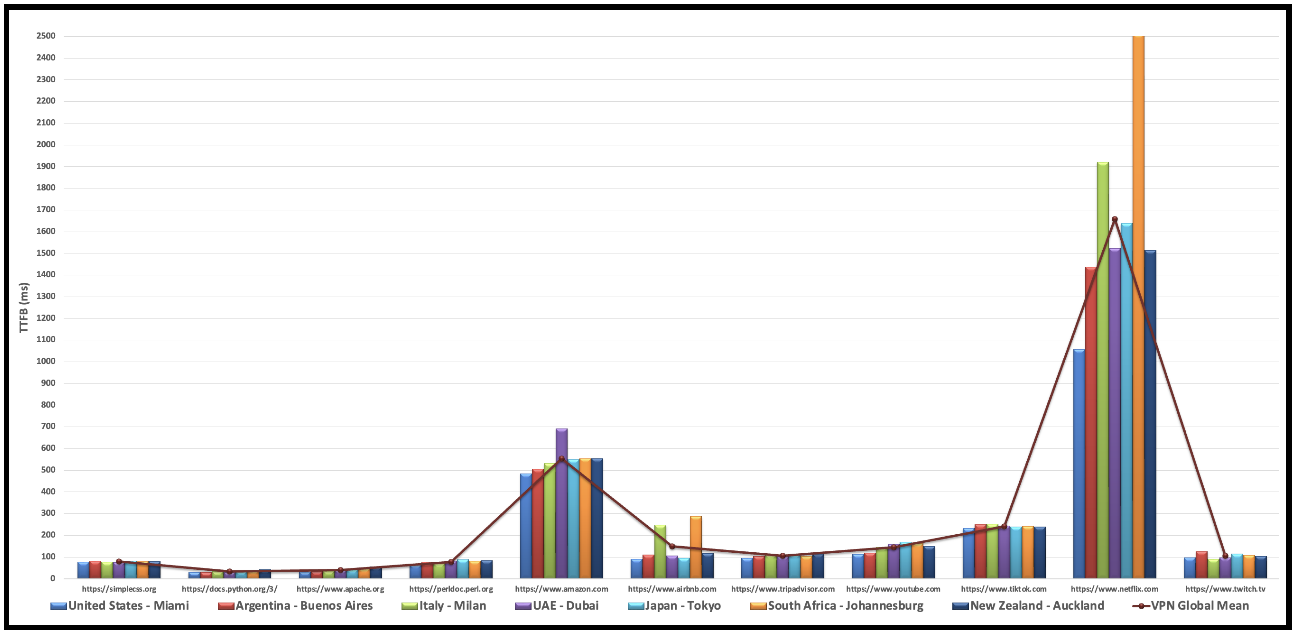


Fig. 7. Time to First Byte by VPN Server Location (Based on Distance from the U.S.)

Website	United States - Miami	Argentina - Buenos Aires	Italy - Milan	UAE - Dubai	Japan - Tokyo	South Africa - Johannesburg	New Zealand - Auckland	VPN Global Mean
https://simplecss.org	77.69 ms	81.19 ms	77.11 ms	74.76 ms	82.02 ms	80.38 ms	79.78 ms	78.99 ms
https://docs.python.org/3/	29.90 ms	28.83 ms	33.07 ms	30.22 ms	33.91 ms	37.05 ms	42.63 ms	33.66 ms
https://www.apache.org	32.62 ms	33.22 ms	36.31 ms	37.99 ms	41.67 ms	45.83 ms	54.65 ms	40.33 ms
https://perldoc.perl.org	64.45 ms	76.81 ms	70.72 ms	82.65 ms	88.53 ms	82.05 ms	84.36 ms	78.51 ms
https://www.amazon.com	485.01 ms	505.60 ms	531.45 ms	691.84 ms	548.91 ms	553.82 ms	555.05 ms	553.10 ms
https://www.airbnb.com	90.21 ms	109.89 ms	247.68 ms	105.23 ms	94.51 ms	286.73 ms	117.22 ms	150.21 ms
https://www.tripadvisor.com	95.20 ms	104.41 ms	110.85 ms	102.36 ms	113.28 ms	104.51 ms	113.60 ms	106.32 ms
https://www.youtube.com	113.46 ms	118.02 ms	145.38 ms	159.38 ms	168.36 ms	162.82 ms	148.71 ms	145.16 ms
https://www.tiktok.com	232.07 ms	249.92 ms	251.34 ms	243.14 ms	238.83 ms	241.97 ms	238.18 ms	242.21 ms
https://www.netflix.com	1057.71 ms	1437.66 ms	1921.31 ms	1522.15 ms	1638.26 ms	2509.51 ms	1512.68 ms	1657.04 ms
https://www.twitch.tv	97.50 ms	126.25 ms	91.36 ms	97.02 ms	114.00 ms	108.58 ms	103.15 ms	105.41 ms

TABLE III
VPN PERFORMANCE BY GEOGRAPHIC LOCATION

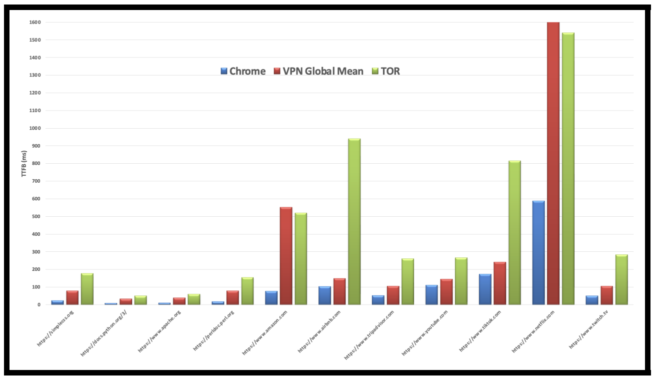


Fig. 8. Time to First Byte (TTFB) by Website and Browsing Configuration

with a global mean of 150.21 ms. Similarly, Tripadvisor's TTFB ranged from 95.2 ms in Miami to 113.6 ms in Auckland, averaging 106.32 ms globally. Amazon showed a TTFB of 485.01 ms in Miami, increasing to 691.84 ms in Dubai, showing higher TTFB values for dynamic content websites in more remote regions.

Video platforms like Netflix had the highest TTFB values, with a global mean of 1657.04 ms. Locations like South Africa and New Zealand experienced significant delays, while Miami generally performed better. These results show that, for most websites, the VPN server located in Miami benefited from its proximity to New Jersey, where the experiments were conducted. However, even with regional differences, VPN performance remained relatively consistent overall.

Website	Chrome	VPN Global Mean	TOR
https://simplecss.org	24.46 ms	78.99 ms	178.62 ms
https://docs.python.org/3/	9.64 ms	33.66 ms	50.07 ms
https://www.apache.org	10.77 ms	40.33 ms	60.92 ms
https://perldoc.perl.org	17.98 ms	78.51 ms	154.78 ms
https://www.amazon.com	77.12 ms	553.10 ms	520.40 ms
https://www.airbnb.com	103.30 ms	150.21 ms	941.34 ms
https://www.tripadvisor.com	52.20 ms	106.32 ms	261.64 ms
https://www.youtube.com	112.20 ms	145.16 ms	267.04 ms
https://www.tiktok.com	173.26 ms	242.21 ms	816.37 ms
https://www.netflix.com	589.12 ms	1657.04 ms	1541.64 ms
https://www.twitch.tv	50.93 ms	105.41 ms	284.00 ms

TABLE IV
TIME TO FIRST BYTE (TTFB) BY BROWSING CONFIGURATION

Figure 8 and Table IV compare the Time to First Byte (TTFB) across three configurations: Chrome (without VPN), VPN Global Mean, and Tor. Chrome consistently shows the lowest TTFB values across all websites due to its direct routing and minimal overhead. VPN introduces additional delay but

Website	**VPN vs Chrome Ratio**	**Tor vs Chrome Ratio**	**Tor vs VPN Ratio**
https://simplecss.org	3.23	7.30	2.26
https://docs.python.org/3/	3.49	5.19	1.49
https://www.apache.org	3.74	5.66	1.51
https://perldoc.perl.org	4.37	8.61	1.97
https://www.amazon.com	7.17	6.75	0.94
https://www.airbnb.com	1.45	9.11	6.27
https://www.tripadvisor.com	2.04	5.01	2.46
https://www.youtube.com	1.29	2.38	1.84
https://www.tiktok.com	1.40	4.71	3.37
https://www.netflix.com	2.81	2.62	0.93
https://www.twitch.tv	2.07	5.58	2.69

TABLE V
TIME TO FIRST BYTE (TTFB) RATIOS BY BROWSING CONFIGURATION

remains significantly faster than Tor, which exhibits the highest TTFB values across all configurations due to its onion routing mechanism. However, for Netflix, VPN performed worse than Tor. This occurred because some VPN servers had slower performance for Netflix, which affected the global mean and made VPN appear slower than Tor in this specific case.

Table V shows the ratios between the three configurations. For example, VPN is, on average, three times slower than Chrome (without VPN), with Amazon being the website where VPN was seven times slower than Chrome. Tor was, on average, 5.7 times slower than Chrome, with Airbnb showing the highest ratio, at nine times slower than Chrome. When compared to VPN, Tor was, on average, 2.34 times slower than VPN. The results indicate that Tor is the slowest because it routes traffic through multiple nodes to provide additional privacy. These added layers of routing significantly increase the Time to First Byte. Therefore, Tor's focus on privacy and anonymity comes at the cost of significantly higher delays than a direct connection like Chrome or even a VPN.

V. CONTRIBUTIONS

We believe each team member contributed equally to the success of this project, ensuring a collaborative effort in both the experimental and analytical phases. Anish was responsible for conducting experiments to measure full page load times, utilizing python code to automate data collection with a headless Chrome and TOR browser, and optimizing that framework for consistent results. Daniel was responsible for testing Time to First Byte (TTFB) using a diverse range of VPNs, as well as Tor and Chrome in normal conditions, to evaluate performance across the world. Daniel developed and executed automated scripts to measure TTFB, analyzed the collected data to understand how these different configurations influence server response times, and created detailed tables and graphs to present the results. Nilang was responsible for checking the latency compared with Chrome browser, chrome + VPN, and TOR browse. Tested with Python code and JavaScript, which helped to get the idea about the difference it makes by using a proxy and running it directly on the browser. Liam was responsible for designing and running the experiments to measure download speeds, again using python code, as well as recording and organizing the associated results. All members participated in data collection, shared responsibilities for cleaning and modeling their metric data, and contributed to writing and organizing the final presentation and report.

VI. CONCLUSIONS

We believe our research highlights the significant performance trade-offs associated with using privacy-enhancing tools like the Tor Browser and VPNs as compared to standard web browsing. Across all measured metrics, page load times, download speeds, network latency, and time to first byte, Tor consistently exhibited the slowest performance, often by a substantial margin. This outcome underscores the inherent limitations of Tor's onion-routing mechanism, which prioritizes anonymity and security at the expense of speed and responsiveness. Though VPNs also introduced some performance degradation, their impact was far less severe than Tor, making them a more practical option for users balancing privacy and usability. These findings align with anecdotal observations that Tor is sluggish, particularly for dynamic and resource-intensive websites. Thus, while Tor remains a crucial tool for secure and anonymous communication, its practical use is limited to scenarios where heightened privacy outweighs the need for performance, such as in environments requiring strong censorship resistance with anonymity. We hope our research provides a quantified perspective on the usability gaps of these tools and insights for users navigating the balance between privacy and browsing efficiency.

VII. APPENDIX

A. Project Code Repository

https://github.com/teelgrass/CS552-project