

Bot or Not? Detecting AI-Driven Web Traffic Using Network Metadata Analysis

Author: [William Hatfield, william.e.hatfield@gmail.com](mailto:william.e.hatfield@gmail.com)

Advisor: [John Hally](#)

Accepted: August 10, 2026

Abstract:

The webscape of today is seeing a surge in automated web traffic from bots, scrapers, and increasingly sophisticated AI browsing agents. Despite the evolving threat, detection engines still rely heavily on either simple single-vector signatures or computationally intensive deep packet inspection, both of which are costly and intrusive for identifying automated traffic. LLM agents, specifically, are better able to mask their traffic with pauses for inference, analysis, and decision-making, rendering single-vector detection ineffective. By combining timing variability, packet burstiness, entropy, and directional ratios into a multi-variable behavioral analysis, this study shows that autonomous traffic can be identified using network metadata alone, with strong differentiators. This detection method is lightweight, non-intrusive, and easily integrated into existing security toolsets, offering defenders a scalable alternative to traditional bot detection and mitigation techniques.

1. Introduction

A majority of today's websites depend on ad revenue, search engine optimization (SEO), and user interaction to survive and thrive. A user discovers a site via a search engine, navigates the site, interacts with elements, is exposed to ads, and maybe posts in a forum or makes a purchase through a store. This traffic boosts the site's metrics and generates revenue, either directly through purchases or indirectly through ad placements. Historically, this traffic and interaction have been primarily driven by humans, serving as the lifeblood of the internet at large. The more people visit a site, the better it does, and the more it grows. If user traffic declines, the site starts losing money, and eventually it shuts down. This calculus has largely held and been relatively easy to gauge via server-side tracking and metrics, allowing site owners to measure the "health" of their site.

Since the early days of the internet, people have been trying to automate browsing and create curated experiences that show only content of interest to them (Balabanović & Shoham, 1995). From the first basic scrapers to today's advanced agents and agentic browsers that can execute complex tasks on users' behalf, browsing tools have evolved into more sophisticated tools. While these tools may make browsing an increasingly cluttered internet more straightforward for users, their growing prevalence poses a problem for website owners, administrators, and defenders.

1.1. An Evolving Webscape

As previously discussed, websites rely on ad revenue and SEO (Search Engine Optimization) for traffic and revenue generation. Automated browsers, though, shift the paradigm; AI crawlers and the like can scrape tens of thousands of pages without sending a single human visitor to a site, leaving metrics that appear to show increasing visitation but with decreasing ad revenue (Playwire Strategy Team, 2025). AI-driven traffic grows every year, and most of it is for model training, resulting in even fewer user referrals and engagement per visit (Tomé, 2025). These bots and agents also pose security concerns; they can probe sites for weaknesses and actively exploit vulnerabilities (Mudryi et al., 2025; Zhang et al., 2025). Site owners face a challenge: how do I differentiate automated traffic from human traffic?

1.2. A Growing Challenge

As bots and agents grow more advanced, so do their methods of evading and defeating detection strategies (Wiefling et al., 2019). Traditional detection methods, such as user-agent identification, browser fingerprinting, behavior analytics, and deep packet inspection, are resource-intensive, raise privacy concerns, and typically require advanced configuration and monitoring. This paper approaches the challenge with a simpler solution: can bot-, AI-, and human-driven web traffic be differentiated without advanced detection tools by analyzing high-level network metadata? This metadata has long been used to detect malicious traffic in intrusion detection and prevention systems without requiring deep analysis (Sperotto et al., 2010). This work explores the same logic for detecting automated traffic without requiring advanced tools.

2. Research Method

This research uses a controlled web server environment to capture network traffic while three groups of users browse the site: humans, bots, and advanced AI agents. All traffic is captured on the server during each browsing session and analyzed to determine what identifying characteristics, if any, can be found across user groups. The research consists of four components: environment construction, user interaction/traffic generation, metadata capture, and metadata analysis.

2.1. Environment Construction

A simple web server was built on a dedicated Geekom A8 mini-PC with Ubuntu Desktop 26.04 LTS. The build included:

- Python3
- A Flask application to provide a web framework.
- Gunicorn provides the HTTP server and uses worker processes to handle incoming traffic.
- Simple pages linked together, including AI-generated articles and a consent page.
- Downloadable PDF files
- Static and dynamic assets for background traffic.

- A basic search function for interaction.

2.2. Traffic Generation

Three distinct groups of subjects were used to generate traffic for analysis: humans, crawlers and scrapers, and advanced agentic and Large Language Model (LLM)-Driven bots. Humans were from geographically distinct regions, and all bots and agents were run from remote VPN endpoints to simulate real-world network latency of remote connections.

2.2.1. Human Subjects

Human subjects were given a list of discrete actions to take (open an article, run a search, download a file, etc.), followed by two to three minutes of ‘freeform browsing’ where the subjects decided on their own actions. This was used to generate traffic patterns akin to natural browsing sessions with pauses and varying levels of interaction with the site.

2.2.2. Bots

Traditional, simple automated tools such as wget, Invoke-WebRequest, BeautifulSoup, and headless browsers like Playwright and Puppeteer were used to generate “bot-like” traffic. This traffic contains common bot indicators such as tight packet bursts, minimal dwell time, microsecond flows, and predictable spacing. Due to the varying levels of complexity of bot scripts, these results were more varied than anticipated.

2.2.3. AI

Agentic browsers and AI-driven programs, such as agentic Playwright and Selenium, and visual-based LLMs via Browser Use were used to generate traffic intended to mimic human behavior. They were given instructions similar to those for human subjects, with random timing and randomized actions to simulate natural browsing traffic. The common frameworks underlying the AI agents resulted in a homogenous dataset across the AI sessions.

2.3. Metadata Captured and Variables Measured

After each session, the following metadata characteristics were extracted from the capture files:

- Session duration and total packet count
- Total unique flows and average flow lifetime
- Inter-arrival time (IAT) min, max, median, mean, and standard deviation
- Min, max, median, mean, standard deviation, and entropy of packet sizes
- Min, max, median, mean, standard deviation, and entropy of inbound and outbound packets*
- The ratio of outbound-to-inbound packets*
- Packet burstiness, or the measure of how packets arrive: in clumps or uniformly

*Packet directionality measured from the perspective of the client.

2.4. Extraction and Analysis Workflow

Each inbound session was processed using an extraction, analysis, and comparison pipeline to gather the aforementioned variables, build master datasets for each subject category, and conduct comparative analyses.

2.4.1. Capture, Extraction, and Normalization

A Python script using the scapy module captured all inbound traffic on port 8000. It identified unique sessions by combining the source IP with the source and destination port numbers, and auto-created new PCAP (packet capture) files for each session, which were set to time out after 10 minutes of inactivity. Each session PCAP was then run through a shell script that used tshark to extract time, 5-tuple, and size for every packet and create a CSV (comma-separated value) spreadsheet organizing the individual flows of that session.

2.4.2. Metadata Analysis and Datasets

Each CSV was then processed with a series of Python analysis scripts using the pandas and numpy modules to analyze packets and flows and calculate the metadata variables for comparison. The output of these scripts was then combined into one master dataset for each of the three subject categories.

2.4.3. Comparison and Plotting

These master datasets were stripped of source IPs. They were then run through a comparative analysis using a Python script with pandas, numpy, scipy, and matplotlib to examine behavioral patterns, temporal trends, and packet distributions and identify factors that could be used to differentiate between subject groups using only the available metadata. Finally, a script using pandas, numpy, matplotlib, and scikit-learn was used to perform principal component analysis (PCA) and plot the results for a 2D visualization of the behavioral differentiators.

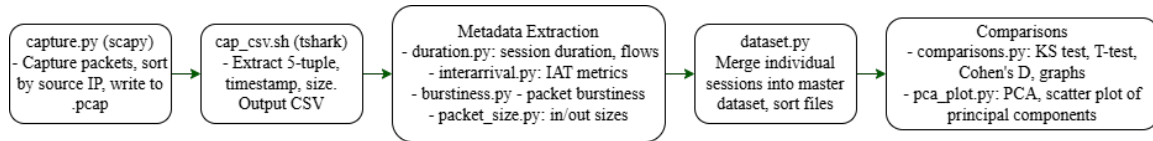


Figure 1: Data Analysis Flow

3. Findings and Discussion

3.1. Feature-Level Analysis

Each metadata feature extracted from the sessions underwent comparative analysis to assess feature separability and determine whether the three subject groups (Humans, Bots, AIs) were sufficiently differentiated to reliably classify traffic as belonging to one of the groups without packet inspection. Each pairing of the groups (Humans vs Bots, Humans vs AI, Bots vs AI) was used with three statistical analyses to determine the following:

- Kolmogorov-Smirnov (KS) statistic: this measures how different two given distributions are. This tells us how different the ‘behavior’ of the two groups is. A value of 1 indicates no overlap between the two groups.
- t-test: This measures the difference in the average values between the groups. This tells us whether one group tends to have higher or lower values than the other.
- Cohen’s d: This measures the significance of the difference between the values. This tells us whether the difference is statistically significant. Anything above ~0.8 indicates a statistically significant difference.

The following sections will discuss the results of the feature-level analysis and their implications.

3.1.1. Inter-Arrival Time

Inter-arrival time (IAT) measures the time between packets in a given flow. Humans have irregular, widely spaced timing due to natural pauses in reading and scrolling, as well as distractions. Bots and AI, on the other hand, tend to have regular, uniform timing.

The high values for humans vs AIs indicate that this is a strong differentiator between the two groups. There is no overlap between the two groups, and the difference is statistically significant. Humans vs bots also exhibit a large degree of separation in distribution but very little difference in mean values, so this is not a statistically significant feature for that pairing. Bots vs AI have a moderate separation but enough overlap in mean values that the difference is not statistically significant.

Comparing Feature: IAT Mean					
KS Test		t-test		Cohen's d	
H vs B	0.8182	H vs B	0.2235	H vs B	0.0953
H vs A	1	H vs A	4.8689	H vs A	2.3077
B vs A	0.3636	B vs A	1.0389	B vs A	0.4458

Table 1: IAT Mean Comparison

Again, humans vs AI shows no overlap and a large statistical differentiation in table 2. There is also a significant difference between humans and bots, though, not as large.

Comparing Feature: IAT StD					
KS Test		t-test		Cohen's d	
H vs B	0.9091	H vs B	1.9012	H vs B	0.8107
H vs A	1	H vs A	3.2012	H vs A	1.3982
B vs A	0.3636	B vs A	0.4331	B vs A	0.1883

Table 2: IAT Standard Deviation Comparison

3.1.2. Burstiness

This feature measures how “clumped up” the packets of a session are. Humans tend to be bursty, with pauses followed by bursts of activity, while bots tend to be more uniform and machine-like, and AIs are in the middle.

The high Cohen’s d values across the board in Table 3 indicate that burstiness is a strong indicator for all three subject groups, with each group having its own distinct browsing rhythm. Humans have organic bursts as they browse, bots follow scripted actions with minimal time difference, and AIs, even when told to mimic humans, still produce structured bursts due to the underlying logic and decision loops driving the traffic.

Comparing Feature: Burstiness					
KS Test		t-test		Cohen's d	
H vs B	0.7273	H vs B	3.3588	H vs B	1.4322
H vs A	0.4545	H vs A	1.6673	H vs A	0.8676
B vs A	0.5758	B vs A	2.563	B vs A	1.1233

Table 3: Burstiness Comparison

3.1.3. Session Duration

This feature measures the duration of a session in seconds. This is the single strongest indicator of human vs. machine browsing, with massive Cohen’s d values for both machine groups. Humans have a wide range of browsing times with pauses for reading, thinking, and distractions. Bots have very short durations on average, completing tasks and sessions as quickly as the underlying logic allows. AIs, even when instructed to pause and delay like a human, still operate at machine speed and do not take pauses as long as humans do.

Comparing Feature: Session Duration (Seconds)					
KS Test		t-test		Cohen's d	
H vs B	0.9091	H vs B	6.1784	H vs B	2.6345
H vs A	1	H vs A	5.6087	H vs A	2.4638
B vs A	0.5455	B vs A	1.1943	B vs A	0.574

Table 4: Session Duration Comparison

3.1.4. Packet and Flow Counts

This set counted the total number of packets and the number of unique flows in each session.

While there was moderate overlap in packet counts between humans and AIs, the spread of the numbers is considered statistically significant enough to serve as a strong indicator. There were large variances in traffic counts for the bot group due to the varying complexity of the scripts used, with some scripts only fetching basic assets or performing surface-level scans, leading to this being a weaker differentiator for bots.

Comparing Feature: Total Packets					
KS Test		t-test		Cohen's d	
H vs B	0.5455	H vs B	0.1852	H vs B	0.079
H vs A	0.4545	H vs A	2.0451	H vs A	0.9719
B vs A	0.5455	B vs A	0.6332	B vs A	0.2713

Table 5: Total Packets Comparison

As with total packets, there was moderate overlap across the three groups in flow counts, but the variance was sufficient to serve as a good signal of human vs. AI browsing. Moreover, varying levels of complexity in bot scripts lead to a large variation in flow counts across bots.

Comparing Feature: Total Unique Flows					
KS Test		t-test		Cohen's d	
H vs B	0.6364	H vs B	1.5354	H vs B	0.6547
H vs A	0.4848	H vs A	2.046	H vs A	1.0406
B vs A	0.6364	B vs A	0.4526	B vs A	0.2006

Table 6: Unique Flow Comparison

3.1.5. Flow Lifetime

The flow lifetime measures the average lifetime of flows across the session. With moderate KS values and smaller Cohen's d values, flow lifetime is not a strong indicator for traffic from the three groups. Flow lifetime is more a function of the website and the server it is hosted on, with server-side performance affecting asset retrieval speed more than client-side attributes.

Comparing Feature: Avg Flow Lifetime

KS Test		t-test		Cohen's d	
H vs B	0.3636	H vs B	1.3933	H vs B	0.5941
H vs A	0.3333	H vs A	0.5022	H vs A	0.2791
B vs A	0.5758	B vs A	0.7692	B vs A	0.3637

Table 7: Average Flow Lifetime Comparison

3.1.6. Aggregated Features

Aggregated features measure the characteristics of the sizes of all packets in a session.

In general, outbound packets (from the perspective of the client) are smaller as they are mostly GET requests, while inbound packets, the responses and requested assets, are larger. Humans, who ‘consume’ all the content of a web page, tend to have larger averages while bots, who only fetch required assets for their tasks, have smaller averages. This serves as a medium differentiator between humans and AIs, with averages typically between bots and humans, with no outliers.

Comparing Feature: All Packets Mean					
KS Test		t-test		Cohen's d	
H vs B	0.6364	H vs B	0.2638	H vs B	0.1125
H vs A	0.7273	H vs A	3.2129	H vs A	1.4066
B vs A	0.6364	B vs A	0.8211	B vs A	0.3512

Table 8: Mean of Packet Sizes Comparison

Another weak indicator, table 9 shows that humans and bots had large variability in their packet-size standard deviations, while AI values were almost identical, right in the middle of the other two groups.

Comparing Feature: All Packets StD					
KS Test		t-test		Cohen's d	
H vs B	0.3636	H vs B	0.4722	H vs B	0.2013
H vs A	0.6364	H vs A	1.0723	H vs A	0.459
B vs A	0.7273	B vs A	1.2306	B vs A	0.5256

Table 9: Standard Deviation of Packet Sizes Comparison

The randomness of packet sizes in a session, indicating how many packet sizes were observed, shown in table 10. This serves as an indicator of human vs. AI traffic due

to the low variance in AI entropy values, resulting in a large mean difference and a correspondingly large Cohen's value, indicating structured traffic patterns.

Comparing Feature: All Packets Entropy					
KS Test		t-test		Cohen's d	
H vs B	0.4545	H vs B	2.4284	H vs B	1.0355
H vs A	0.6364	H vs A	3.3667	H vs A	1.4968
B vs A	0.3636	B vs A	0.2689	B vs A	0.1196

Table 10: Entropy of Packet Sizes Comparison

3.1.7. Outbound vs Inbound

These features provide measures of various characteristics between inbound and outbound packets, from the perspective of the client.

Table 11 shows the spread of transmitted packet sizes from the client. AI is the outlier here, with a very narrow range of values, making it a good indicator to differentiate it from the other two groups.

Comparing Feature: Outbound Packets StD					
KS Test		t-test		Cohen's d	
H vs B	0.3636	H vs B	0.2045	H vs B	0.0872
H vs A	0.7273	H vs A	2.7608	H vs A	1.1897
B vs A	0.7273	B vs A	1.9185	B vs A	0.8213

Table 11: Standard Deviation of Outbound Packets Comparison

One of the stronger indicators of bot vs. AI is the randomness of outbound packet sizes as seen in Table 12. Bots had high entropy, similar to humans, due to the different complexities of the various tools used in the bot category. All AI sessions were consistent, though, with all entropy values falling within .4 of each other.

Comparing Feature: Outbound Packets Entropy					
KS Test		t-test		Cohen's d	
H vs B	0.5455	H vs B	1.1454	H vs B	0.4884
H vs A	0.3636	H vs A	0.9463	H vs A	0.4133
B vs A	0.8182	B vs A	2.232	B vs A	0.968

Table 12: Entropy of Outbound Packets Comparison

One of the top human vs AI indicators, the “Bytes Ratio” feature, shown in Table 13, shows outbound bytes divided by inbound bytes. High values show asset-heavy content being received by the client, typical of humans. Humans also had a wide range of values, indicating a variety of client architectures. AI had medium ratios with a small spread; many AI systems run on similar backend frameworks, leading to similar patterns in packet directionality.

Comparing Feature: Bytes Ratio Out:In					
KS Test		t-test		Cohen's d	
H vs B	0.5455	H vs B	0.4149	H vs B	0.1769
H vs A	0.7273	H vs A	3.9463	H vs A	1.7435
B vs A	0.4545	B vs A	1.1807	B vs A	0.5041

Table 13: Outbound:Inbound Bytes Comparison

3.1.8. Separability Ranking

The cumulative Cohen's d and KS values from each feature, across all three pairings, were calculated then sorted by total Cohen's d values to rank the features by how effective they are at differentiating the groups.

Table 14 shows that metadata features with time and packet-transmission-pattern characteristics are among the strongest differentiators between the groups. High cumulative Cohen's d values indicate large differences between the means across all three group pairings. High cumulative KS statistics indicate noticeable differences in the distributions of values. The very high value of session duration in both measures shows that it would be a key factor in any classification system; machines, even complex ones with complex instructions, cannot replicate the large variability in session duration that humans exhibit.

Separability Ranks, Sorted by Cohen's d Values		
Feature Name	Cum. Cohen's d	Cum. KS Stat
Session Duration (Seconds)	5.6724	2.4545
Burstiness	3.4231	1.7576
IAT Mean	2.8489	2.1818
All Entropy	2.652	1.4545

Bytes Ratio Out:In	2.4246	1.7273
IAT StD	2.3971	2.2727
Outbound StD	2.0982	1.8182
Total Unique Flows	1.8959	1.7576
All Packets Mean	1.8703	2
Outbound Packets Entropy	1.8698	1.7273
Inbound Packets Entropy	1.5727	1.2727
Inbound Packets StD	1.3854	1.2879
Total Packets	1.3222	1.5455
Average Flow Lifetime	1.237	1.2727
All Packets StD	1.186	1.7273
Outbound Packets Mean	0.9253	1.5606
Inbound Packets Mean	0.8413	1.1061

Table 14: Separability Ranking of Features

Taken together, the feature comparisons show strong evidence that the characteristics of different groups' browsing sessions: the high variability of humans, varied but rigid values of bots according to the complexity of their instructions, and the uniformly consistent, pattern-based values of the AI sessions, can be used to classify the sessions accurately.

3.2. Top Differentiators

Of the 17 features analyzed across sessions and compared between groups, the top five characteristics, all involving temporal and behavioral patterns, stand out as key differentiators between group pairings. The following subsections will analyze the Kernel Density Estimate (KDE) graphs for the three groups across the sessions. A KDE graph shows the probability of a given value occurring across the dataset. The higher the line, the more concentrated that range of values is across the dataset.

3.2.1. Session Duration

Human browsing sessions span a wide, relatively shallow range of durations, from five to over twenty minutes, which is consistent with earlier suppositions that human cognitive processes, reading speeds, and delayed actions cause high variance in browsing session duration. Automated sessions, on the other hand, show tight duration ranges, consistent with programmatic processing and execution loops. AI specifically shows a high peak and tight grouping with little human overlap, demonstrating that the

boundaries set by underlying frameworks and processing speeds prevent long sessions entirely.

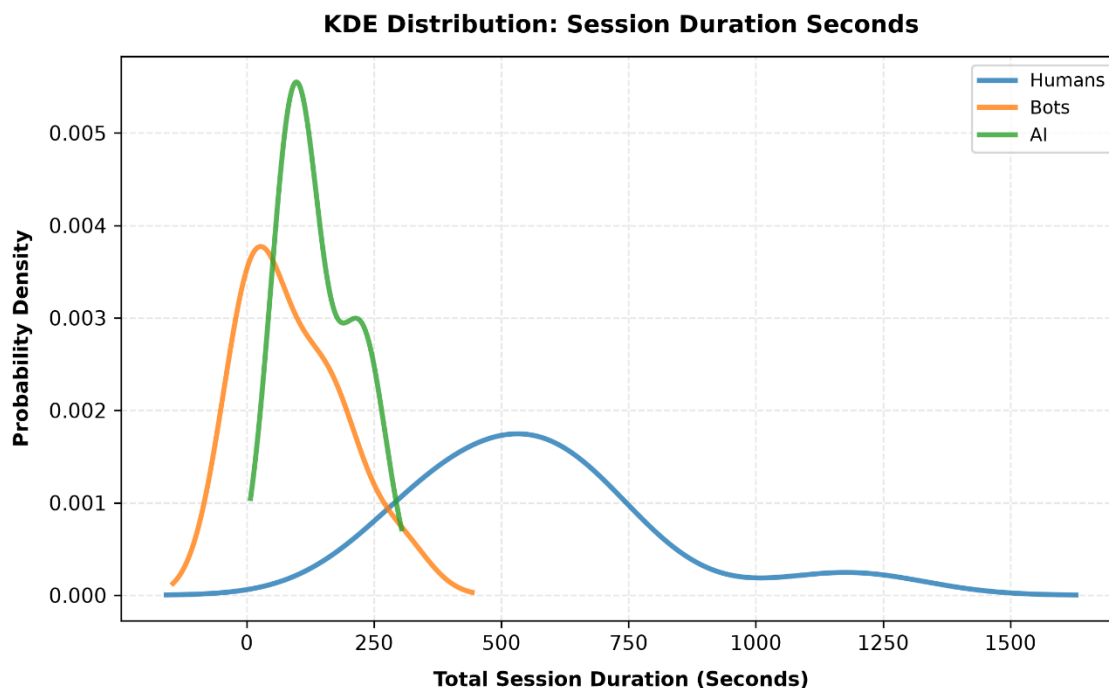


Figure 2: Session Duration KDE

3.2.2. Burstiness

While burstiness has traditionally been used to detect automated traffic generation (Chen et al., 2006; Hilaire et al., 2010), advances in AI capabilities, including vision-based processing, and computational delays in instructions have introduced artificial idle states in sessions. This leads to traffic burstiness very similar to that of humans, as shown in Figure 3. While still a good indicator of continuous-stream bot activity, AI can blend into human traffic patterns when viewed in isolation.

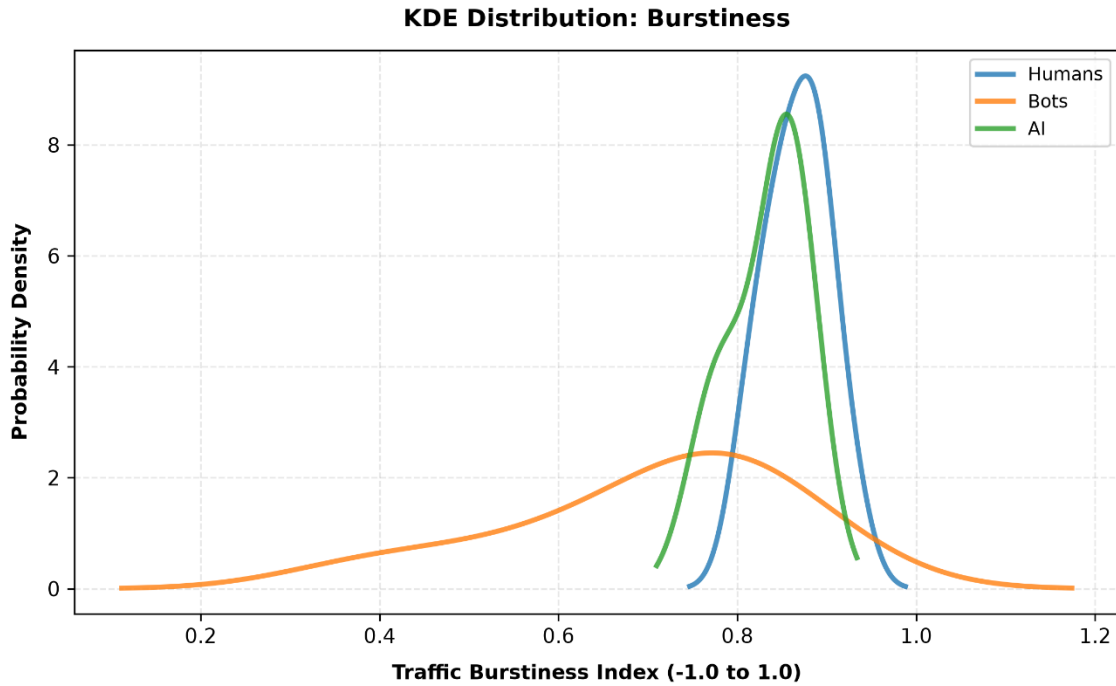


Figure 3: Burstiness KDE

3.2.3. Inter-Arrival Time Mean

While advanced AI agents can blend into human traffic at a macro level through packet burst timing, examining micro-level inter-packet timing reveals very little overlap between the two groups' network behavior. An AI framework's highly optimized network processing results in massive bursts of microsecond-spaced packets, shifting the IAT mean noticeably closer to 0 and away from human standard timing.

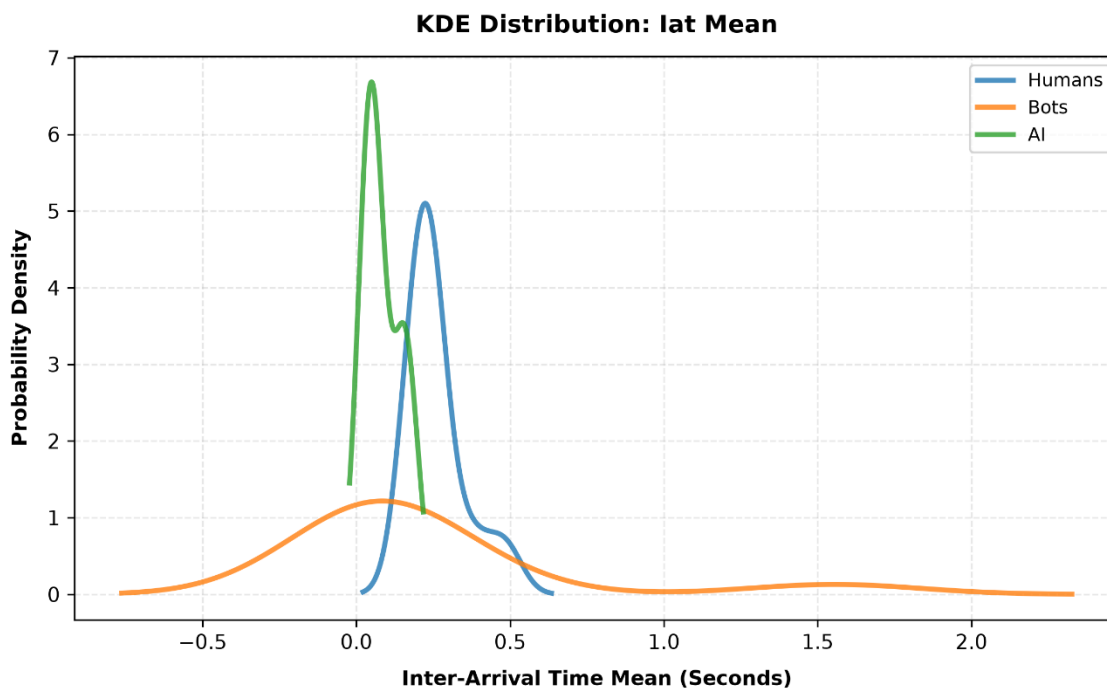


Figure 4: IAT Mean KDE

3.2.4. All Packets Entropy

With high Cohen's d values across both human pairings, total entropy serves as a good general indicator of human vs autonomous traffic. While AIs and bots varied in density, the underlying behavior of sending structured requests and receiving code, elements, and arrays revealed similarities among the automated subjects, setting them apart from the human subjects.

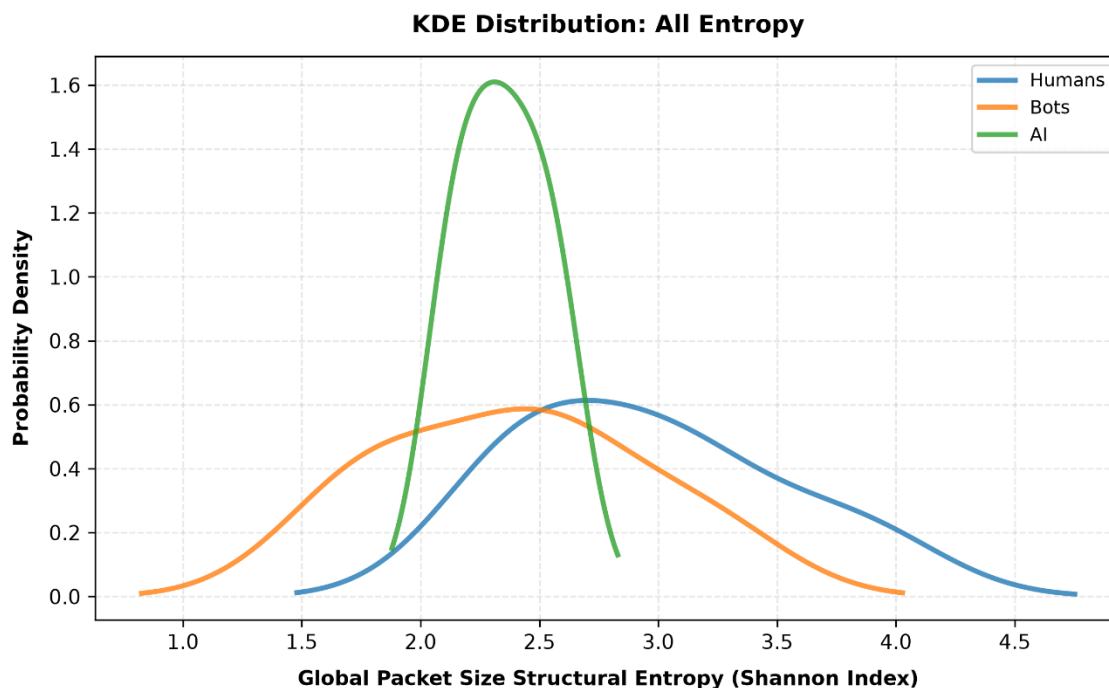


Figure 5: All Packets Entropy KDE

3.2.5. Outbound:Inbound Byte Ratio

The bot sessions' widely varying ratio values, a result of varying levels of complexity in the scripts and tools used, make the out:in ratio a weak indicator for distinguishing them from the other groups. However, the AIs' tightly compressed, dense ratio values, a result of the underlying logic loops executing the same actions, requests, and retrievals for every page accessed, set them apart from humans, whose behavioral variability and uneven browsing trends are evident.

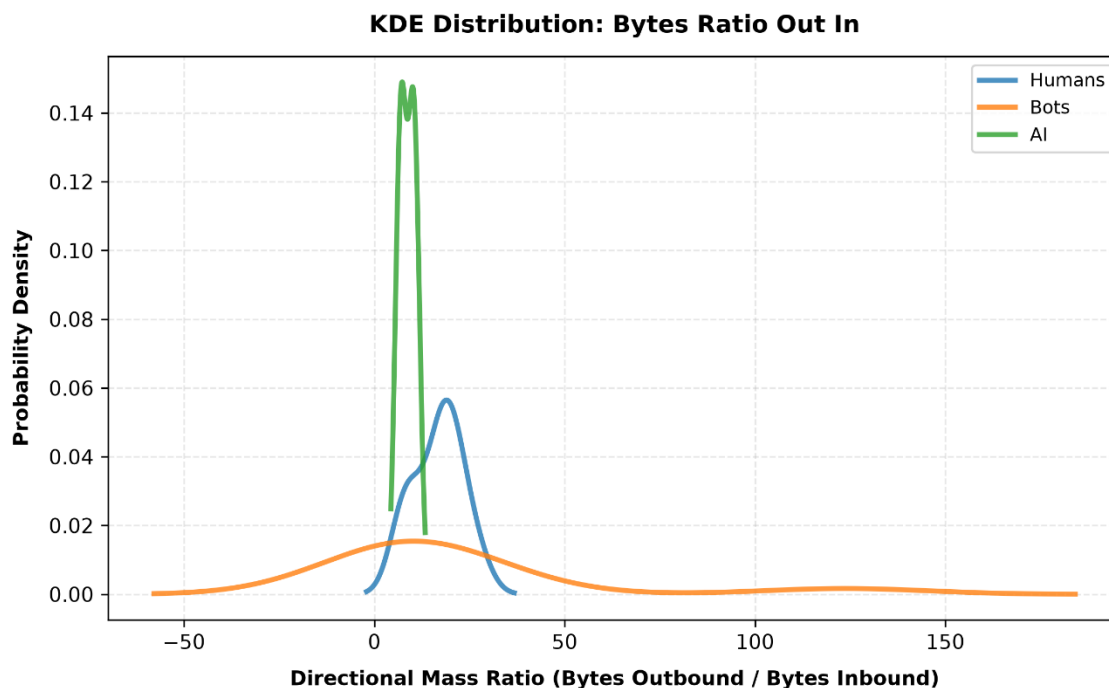


Figure 6: Bytes Out:In Ratio KDE

3.3. Principal Component Analysis

Shown in Figure 7, PCA (Principal Component Analysis) plots take many correlated variables and condense them into a set of “principal components” that can be represented in a plot. This plot uses 17 features across 28 datasets, identifies the key variables, and compresses their representation onto two axes to demonstrate how they affect overall variance.

PC1, the “Volumetric Vector Axis”, represents volume-based variables: packet counts, packet sizes, and packet ratios, and accounts for 34.9% of variance. PC2, the “Temporal Variance Axis”, represents time-based variables: session duration, burstiness, and inter-arrival times, and accounts for 18.2% of variance. The labeled vector arrows indicate how each key variable affected the plotting of the sessions. The two components, representing 5 variables, accounted for 53.1% of the total variance, indicating that these 5 variables alone summarize much of the difference between the subject groups and can be used to identify them.

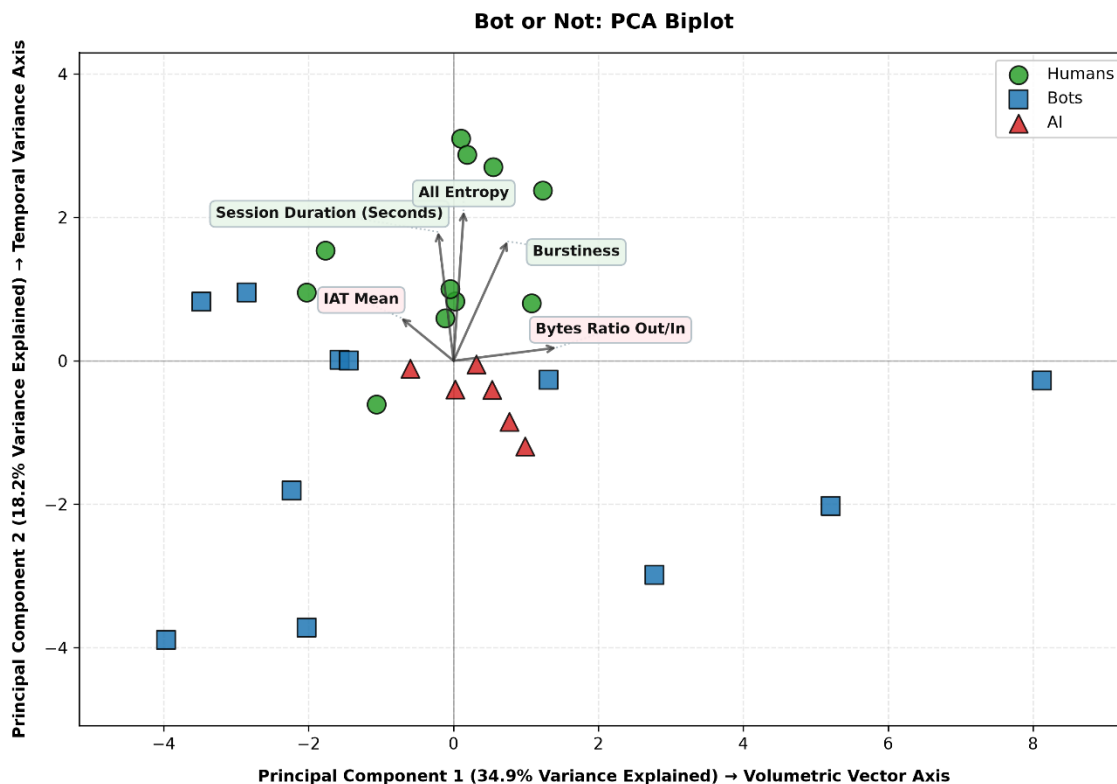


Figure 7: PCA Plot

3.3.1. Interpretation

Human sessions form a dense cluster in the upper half of the plot, concentrated around 0 on the volumetric axis and distributed primarily between 0.5 and 3.0 on the temporal axis. This demonstrates that humans maintain consistency across packet counts and sizes while maintaining random pacing patterns and packet payload entropy.

AI sessions trended in the other direction with a volumetric variance between -0.5 and 1.0 and a temporal variance between -0.1 and -1.2. This highly dense clustering provides evidence that AI logic loops act with machine-like consistency across all runs, even when using different agents and processing systems.

With a diverse set of scripts, tools, and services comprising the bot group, their plots are scattered across both principal components, failing to maintain any real consistency beyond a general grouping around 0 on the temporal axis. From simple single-thread, low-packet tools such as Invoke-WebRequest and opengraph.io to multi-

threaded mass-transmission tools such as concurrent wget loops and GTMetrix processing, the dispersion along the volumetric axis showcases the group's diversity.

Ultimately, the PCA plot shows that only the top 5 of the 17 analyzed variables can differentiate sessions between the groups when only network metadata is available. While AI agents can emulate human pacing with “think time,” burstiness, and transmission ratios, metrics such as entropy and IAT values expose the underlying automation. Neither autonomous group can maintain random session durations as humans can, and the universally low IAT values across all autonomous sessions separate both groups from the human group.

Class	Marker	Typical Region	Behavioral Character
Human	Green circle	Upper-central region	Irregular, high-entropy, inbound-heavy
Bot	Blue square	Spread laterally across both sides	Scripted, repetitive, mechanically bursty
AI	Red triangle	Center-lower region	Structured, outbound-heavy, model-driven

Table 15: Visual Grouping of Plot

4. Recommendations and Implications

4.1. Recommendations for Practice

4.1.1. Multi-Vector Behavioral Analysis and Baselines

While traditional intrusion detection and intrusion prevention systems (IDS/IPS) can detect bot traffic through analysis of session burstiness and volumetric monitoring, LLM-based AI agents can effectively evade detection by mimicking human burstiness with “think time” and inference delays. By designing machine-learning traffic classifiers that combine volumetric and temporal variables (e.g., Random Forest or Gradient-Boosted Decision Trees), site owners and defenders can more effectively identify autonomous traffic.

At the same time, human variability, randomness, and burstiness can be used to build environment- or user-specific baselines. Once baselines are established, deviations

in packet timing, traffic ratios, average session lengths, and other metrics can serve as detection signals for non-human traffic.

Implementation of these classifiers and baselines as part of metadata-first triage pipelines at network boundaries or load balancers can enable early detection of autonomous traffic. This can reduce the need for packet inspection, reducing computation needs and alleviating privacy concerns.

4.1.2. AI-Centric Security

Bot detection and mitigation have historically been seen as bandwidth-conserving measures or basic data-loss prevention tools. The rising prevalence of AI agents traversing the web (Yang et al., 2025; Ning et al., 2025) reflects an evolution in the threat landscape facing defenders. AI agents can quickly scrape backend logic and structure, evaluate vulnerabilities on the fly, modify input based on received errors to achieve desired results, and extract large amounts of information (Mudryi et al., 2025; Zhang et al., 2025). Shifting focus and adding capabilities to detect and respond to these threats quickly and efficiently will become more vital as AI agents and their behaviors evolve and become more ingrained in web usage.

4.2. Implications for Future Research

4.2.1. Evolving AI Agents Mimic Human Variability

As AI agents become more complex, they will get better at mimicking human browsing patterns and behaviors. By artificially altering burstiness, inserting larger micro-delays between packets, and implementing longer inference and think-time thresholds, more advanced agents can more effectively mask themselves among human browsers. Future research would benefit from a greater focus on AI behavioral indicators and patterns. By using a more varied set of frameworks, agents, and models during capture and analysis, a large, representative sample of AI tools can be obtained, enabling detection frameworks and engines to be refined and improved as the threat evolves.

4.2.2. Multi-Agent Ecosystems

As individual agents become more complex, the evolution of multi-agent collaboration will also alter the signals and indicators of AI traffic. By leveraging multi-

agent traffic flows, inter-agent coordination, and distributed processing, multi-agent frameworks can operate differently from single-agent tools. Research exploring these frameworks and identifying key characteristics in their traffic flows can complement the refinement of single-agent detection and provide a more holistic detection framework.

4.2.3. Anti-AI Web Elements

Visually based LLM agents rely heavily on CSS (Cascading Style Sheets) and DOM (Document Object Model) layouts and structures to inspect sites and make action and navigation decisions. Further research on the practicality of “adversarial layouts”: shifting coordinates, invisible anchor links, and small random changes in logic to detect and defeat autonomous traffic sources can be used to counter these agents and models.

5. Conclusion

The increasingly prevalent and increasingly advanced autonomous bots and LLM-based AI agents in today's webscape pose new and evolving challenges to website owners and defenders. From affecting ad revenue to analyzing and exploiting vulnerabilities on the fly, their impact is wide and varied. Traditional detection methods heavily rely on advanced inspection techniques and computationally intense deep packet inspection. This study set out to identify what, if any, characteristics in high-level network metadata captured with lightweight tools can be used to identify and classify autonomous browsing tools.

By capturing and analyzing session metadata, it was shown that temporal and volumetric metadata, such as IAT metrics, packet burstiness and entropy, and transmission ratios, consistently set humans apart from autonomous agents. When these features are combined in a behavioral analysis, humans, bots, and AI agents were shown to occupy distinct regions of behavioral space, further demonstrating that metadata-only detection is feasible and effective.

Existing analysis frameworks, such as Zeek, Suricata, SciPy, and NumPy, are freely available and can be easily integrated into security toolsets to develop metadata-based detection frameworks that effectively detect, isolate, and respond to unwanted autonomous traffic.

References

- Aviad Kaiserman, G. C. (2025, November 7). *AI Agent Detection: A Guide to Identifying Autonomous Traffic*. HUMAN Security.
<https://www.humansecurity.com/learn/blog/ai-agent-signals-traffic-detection/>
- Balabanović, M., & Shoham, Y. (1995). 1995 AAAI Symposium. In *Learning Information Retrieval Agents: Experiments with Automated Web Browsing* (pp. 13–18). Menlo Park, CA; AAAI Press. Retrieved 2025, from
<https://cdn.aaai.org/Symposia/Spring/1995/SS-95-08/SS95-08-003.pdf>.
- Bartels, R. (2025, April 4). *How Cloudflare Detects Bot Traffic | A Deep Dive into Bot Detection Techniques*. Hub & Spoke. <https://hubandspoke.amastelek.com/how-cloudflare-detects-bot-traffic-a-deep-dive-into-bot-detection-techniques>
- Chen, K.-T., Jiang, J.-W., Huang, P., Chu, H.-H., Lei, C.-L., & Chen, W.-C. (2006). Identifying MMORPG Bots: A Traffic Analysis Approach. *Proceedings of the 2006 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology*, 4–12. <https://doi.org/10.1145/1178823.1178829>
- DeepWiki. (2025, December 24). *BOT detection and evasion guide: Digitalmethodsinitiative/browser-fingerprinting*.
<https://deepwiki.com/digitalmethodsinitiative/browser-fingerprinting/2-bot-detection-and-evasion-guide>
- Hilaire, S., Kim, H., & Kim, C. (2010). How to deal with Bot Scum in MMORPGs? *2010 IEEE International Workshop Technical Committee on Communications Quality and Reliability (CQR 2010)*, 1–6. <https://doi.org/10.1109/cqr.2010.5619911>
- Mudryi, M., Chaklosh, M., & Wójcik, G. (2025, May 19). *The Hidden Dangers of Browsing AI Agents*. arXiv. <https://doi.org/10.48550/arXiv.2505.13076>
- Ning, L., Liang, Z., Jiang, Z., Qu, H., Ding, Y., Fan, W., Wei, X., Lin, S., Liu, H., Yu, P. S., & Li, Q. (2025). A Survey of WebAgents: Towards Next-Generation AI Agents for Web Automation with Large Foundation Models. *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, 6140–6150. <https://doi.org/10.1145/3711896.3736555>

- Playwire Strategy Team. (2025, December 8). *How AI Crawling Affects Your Ad Revenue: A Data-Driven Analysis*. Playwire.
<https://www.playwire.com/blog/how-ai-crawling-affects-your-ad-revenue-a-data-driven-analysis>
- Sperotto, A., Schaffrath, G., Sadre, R., Morariu, C., Pras, A., & Stiller, B. (2010). An overview of IP flow-based intrusion detection. *IEEE Communications Surveys & Tutorials*, 12(3), 343–356.
<https://doi.org/10.1109/surv.2010.032210.00054>
- Tomé, J. (2025, August 29). *The Crawl-to-Click Gap: Cloudflare Data on AI Bots, Training, and Referrals*. Cloudflare Blog. <https://blog.cloudflare.com/crawlers-click-ai-bots-training/>
- Vastel, A. (2025a, July 30). *From LLM Scrapers to AI agents: Mapping the AI bot landscape for detection teams*. Security Boulevard.
<https://securityboulevard.com/2025/07/from-llm-scrapers-to-ai-agents-mapping-the-ai-bot-landscape-for-detection-teams/>
- Vastel, A. (2025b, August 5). *From detection to trust: The Evolving Challenge of AI Bot Authentication*. Security Boulevard. <https://securityboulevard.com/2025/08/from-detection-to-trust-the-evolving-challenge-of-ai-bot-authentication/>
- Wiefling, S., Gruschka, N., & Lo Iacono, L. (2019). Even Turing Should Sometimes Not Be Able to Tell: Mimicking Humanoid Usage Behavior for Exploratory Studies of Online Service. *Lecture Notes in Computer Science*, 188–203.
https://doi.org/10.1007/978-3-030-35055-0_12
- Yang, Y., Ma, M., Huang, Y., Chai, H., Gong, C., Geng, H., Zhou, Y., Wen, Y., Fang, M., Chen, M., Gu, S., Jin, M., Spanos, C., Yang, Y., Abbeel, P., Song, D., Zhang, W., & Wang, J. (2025, July 28). *Agentic Web: Weaving the Next Web with AI Agents*. arXiv.org. <https://doi.org/10.48550/arXiv.2507.21206>
- Zhang, K., Tenenholtz, M., Polley, K., Ma, J., Yarats, D., & Li, N. (2025, November 25). *BrowseSafe: Understanding and Preventing Prompt Injection Within AI Browser Agents*. arXiv. <https://doi.org/10.48550/arXiv.2511.20597>

Appendix A: Session Data

session_id	H-01	H-02	H-03	H-04	H-05	H-06	H-07	H-08	H-09	H-10	H-11
session_duration	529.13	288.68	615.84	558.93	393.7	390.08	619.09	714.06	1182.01	629.11	395.51
total_packets	2107	1653	2408	2841	2024	1528	2687	2940	2458	1689	2006
unique_flows	92	85	143	134	113	91	69	133	92	88	75
avg_flow_lifetime	15.74	14.74	22.85	3.32	17.69	8.99	37.81	2.56	2.62	22.88	2.5
iat_mean	0.25	0.17	0.26	0.2	0.19	0.26	0.23	0.24	0.48	0.37	0.2
iat_std	3.34	2.18	2.65	3.29	2.19	2.39	3.13	4.39	12.81	6.79	3.26
iat_min	0	0	0	0	0	0	0	0	0	0	0
iat_max	90.28	57.97	45.03	128.41	45.03	55.98	86.07	136.68	600.99	214.7	95.63
iat_median	5.5E-05	2.8E-05	3.3E-05	2.2E-05	4.6E-05	5.8E-05	3.2E-05	1.9E-05	2.2E-05	2.4E-05	1.2E-05
all_mean	1265.69	1266.84	1468.49	743.18	1595.58	1916.36	1009.61	713.19	1426.71	648.35	1055.48
all_std	1482.05	2986.93	4624.00	2569.00	4103.23	5078.83	1273.24	2771.31	2828.79	1143.84	3265.69
all_min	54	66	54	54	54	54	54	54	66	54	54
all_max	8814	23118	57056	44954	54074	43854	2836	62330	17442	5410	42394
all_median	474	66	66	66	66	66	60	66	66	66	60
all_entropy	3.18	2.85	3.32	2.49	3.93	3.75	2.37	2.30	2.82	2.49	3.06
inbound_mean	149.35	135.93	148.02	112.54	178.18	150.37	98.62	116.05	115.81	135.35	107.08
inbound_std	204.43	187.31	199.95	144.75	234.25	205.71	142.62	160.49	160.16	190.63	145.18
inbound_min	60	66	60	66	60	60	60	66	66	66	60
inbound_max	764	779	777	660	781	758	773	770	763	791	648
inbound_median	60	66	66	66	60	60	60	66	66	66	60
inbound_entropy	2.12	1.56	1.95	1.35	2.20	1.92	0.99	1.24	1.22	1.50	1.24
outbound_mean	1911.24	2304.60	2793.36	1848.63	2996.27	3385.90	1962.91	1882.23	2915.27	1282.96	2674.54
outbound_std	1518.29	3851.34	6269.29	4027.65	5430.05	6518.37	1232.48	4540.24	3591.18	1467.95	4969.62
outbound_min	54	66	54	54	54	54	54	54	66	54	54
outbound_max	8814	23118	57056	44954	54074	43854	2836	62330	17442	5410	42394
outbound_median	2974	1065	400	74	478	476	2836	74	2962	486	476
outbound_entropy	2.45	3.15	3.69	3.47	3.91	3.69	1.89	3.37	3.40	2.80	3.83
bytes_out:in	22.13	18.48	18.81	9.37	17.02	27.06	19.02	8.28	22.17	7.66	14.63
burstiness	0.86	0.85	0.82	0.89	0.84	0.81	0.86	0.90	0.93	0.90	0.89

Table A1: Human Session Data

session_id	B01	B02	B03	B04	B05	B06	B07	B08	B09	B10	B11
session_duration	59.55	9.82	2.07	158.42	101.02	6.68	7.21	295.81	171.01	13.14	173.65
total_packets	6345	61	14	8455	66	19	671	4212	2773	348	3201
unique_flows	40	3	1	182	1	1	1	149	149	4	163
avg_flow_lifetime	4.20	5.50	2.07	30.45	101.02	6.68	7.21	121.73	23.37	5.02	44.10
iat_mean	0.01	0.16	0.16	0.02	1.55	0.37	0.01	0.07	0.06	0.04	0.05
iat_std	0.08	0.68	0.37	0.39	12.15	1.13	0.08	0.33	0.66	0.41	0.33
iat_min	0	0	9.5E-07	0	2.5E-05	0	0	0	0	0	0
iat_max	1.85	4.52	1.02	25.60	98.02	4.52	0.71	3.68	30.63	6.38	3.63
iat_median	9.5E-07	1.3E-04	1.2E-04	1.7E-05	1.6E-03	3.7E-05	0	5.9E-05	2.3E-05	2.3E-05	3.7E-05
all_mean	1400.23	548.95	318.71	1338.31	467.89	568.53	1258.17	454.11	481.85	4627.15	505.77
all_std	5361.31	914.70	661.41	2176.68	739.95	1021.87	3505.30	2262.20	1669.08	7248.10	2700.65
all_min	54	66	66	54	54	66	66	60	66	66	66
all_max	62330	2838	2493	18890	2481	2838	24682	62330	34818	34530	62330
all_median	66	66	66	66	194	66	66	66	66	389	66
all_entropy	1.74	2.67	1.63	2.40	3.20	1.89	1.98	2.73	2.46	3.23	2.41
inbound_mean	72.04	155.59	175.86	85.25	113.44	115.20	72.06	175.25	136.65	80.24	144.35
inbound_std	38.67	188.11	287.14	90.29	65.39	152.79	32.93	218.49	179.50	79.58	192.16
inbound_min	66	66	66	60	60	66	66	60	66	66	66
inbound_max	360	599	827	636	210	550	331	769	733	568	754
inbound_median	66	66	66	66	60	66	66	66	66	66	66
inbound_entropy	0.34	1.54	1.15	0.77	2.11	0.92	0.76	2.14	1.66	0.44	1.67
outbound_mean	6485.58	983	461.57	2891.78	801.50	1072.22	5599.01	794.74	971.94	8541.87	1021.45
outbound_std	10307.02	1175.47	904.31	2498.43	915.40	1334.51	5779.77	3332.06	2507.76	8043.64	4147.62
outbound_min	54	66	66	54	54	66	66	66	66	66	66
outbound_max	62330	2838	2493	18890	2481	2838	24682	62330	34818	34530	62330
outbound_median	2962	488	66	2962	388	291	2962	70	74	4224	74
outbound_entropy	4.18	2.96	1.66	2.88	2.34	2.11	3.07	2.82	2.88	4.16	2.84
bytes_out:in	23.51	5.73	2.62	27.36	7.51	8.38	21.23	3.71	5.01	123.64	4.96
burstiness	0.79	0.61	0.38	0.91	0.77	0.50	0.75	0.65	0.83	0.83	0.72

Table A2: Bot Session Data

session_id	A01	A02	A03	A04	A05	A06
session_duration	81.57	100.27	109.98	91.19	221.37	229.85
total_packets	2156	1974	2041	1849	1404	1458
unique_flows	74	108	97	78	50	42
avg_flow_lifetime	2.34	2.36	3.74	2.59	50.25	54.75
iat_mean	0.04	0.05	0.05	0.05	0.16	0.16
iat_std	0.33	0.38	0.65	0.76	2.10	1.93
iat_min	0	0	0	0	0	0
iat_max	5.88	6.90	27.05	30.54	59.23	41.88
iat_median	0	0	0	0	0	0
all_mean	950.08	724.06	640.93	739.66	832.64	788.27
all_std	3239.68	3268.82	3229.90	3345.39	3547.41	3457.42
all_min	66	66	66	66	66	66
all_max	44954	62330	62330	62330	62330	62330
all_median	66	66	66	66	66	66
all_entropy	2.16	2.52	2.36	2.59	2.32	2.12
inbound_mean	113.87	149.22	128.59	129.15	111.84	104.45
inbound_std	181.13	231.34	174.82	171.42	152.44	140.16
inbound_min	66	66	66	66	66	66
inbound_max	915	966	761	740	769	769
inbound_median	66	66	66	66	66	66
inbound_entropy	0.93	1.60	1.41	1.57	1.22	1.01
outbound_mean	2800.68	1680.58	1616.04	1926.67	2220.20	2355.04
outbound_std	5357.77	5189.71	5367.05	5549.03	5820.89	5985.39
outbound_min	66	66	66	66	66	66
outbound_max	44954	62330	62330	62330	62330	62330
outbound_median	488	74	74	74	74	74
outbound_entropy	3.62	3.21	3.28	3.50	3.45	3.53
bytes_out:in	11.11	6.77	6.60	7.67	10.31	9.84
burstiness	0.80	0.77	0.85	0.88	0.86	0.85

Table A3: AI Session Data

Appendix B: Sourcing Registry

Source IP_Time*	Tool/Service	Key
136.50.192.87		H01
174.232.47.217		H02
9.134.81.238		H03
159.48.173.5		H04
47.223.95.174		H05
162.203.45.15		H06
72.32.180.177		H07
209.40.123.3		H08
64.139.242.115		H09
172.56.203.1		H10
12.153.205.131		H11
209.40.123.3 225346	Wget Concurrent Scraper	B01
4.155.232.126	GTMetrix Audit Process	B02
184.73.83.196	Opengraph.io	B03
95.217.20.13	Websitecrawler.org	B04
146.20.220.116	Powershell Invoke-WebRequest	B05
4.154.170.206	GTMetrix Filmstrip Process	B06
104.204.187.111	BeautifulSoup Crawler	B07
209.40.123.3 235647	Puppeteer Headless Chromium	B08
104.204.187.111 222949	Playwright Headless Chromium (Search Failed)	B09
4.155.4.251	GTMetrix Main Process	B10
209.40.123.3 221835	Playwright Headful Chromium (Search Fixed)	B11
209.40.123.3 193345	Selenium Agentic Headless	A01
209.40.123.3 222840	Selenium Agentic Headful	A02
209.40.123.3 232455	Playwright Agentic Headful	A03
209.40.123.3 122509	Playwright Agentic Headless	A04
209.40.123.3 142317	Browser Use LLM (Gemini)	A05
209.40.123.3 161633	Browser Use LLM (OpenAI)	A06

*Time if applicable for duplicate source IPs

Table B1: Session Source Registry

Appendix C: Project Repository

All project-related files, analyses, and scripts are publicly hosted at [GitHub - whatfield-sti/BotOrNot: Detecting human or bot web traffic with network metadata analysis](https://github.com/whatfield-sti/BotOrNot: Detecting human or bot web traffic with network metadata analysis). · [GitHub](https://github.com).

The screenshot shows the README of a GitHub repository. The main content area on the left contains the following text:

Bot Or Not? Detecting AI-Driven Web Traffic Using Network Metadata Analysis

This repository contains files and information related to the SANS Technology Institute research project for identifying human or bot internet traffic with only high-level network metadata.

The folders contain all the settings and scripts used in the course of the research:

web_server: All files and configurations used for the web server itself. Flask and Gunicorn were used to host and run the server.

capture_extract: -capture.py - Capture incoming traffic to the web server. Create new .pcap for each new source IP with a timeout of 10 minutes. -cap_with_len_csv.sh - Convert output .pcap to CSV for analysis. -burstiness.py - Calculate burst metrics from input session. -duration.py - Calculate session duration metrics from input session. -interarrival.py - Calculate interarrival time metrics from input session. -packet_size.py - Calculate packet size metrics from input session. -all_analysis.py - Run all 4 analysis scripts. -dataset.py - Combine the output of all analysis scripts into a single CSV then move all component files into a subdirectory based on the initial capture name.

analyze_compare: -group.py - Short script to combine all output CSVs from dataset.py into a single master dataset for the given group. -comparisons.py - Run comparison analyses between the 3 master dataset files to calculate KS values, t-test values, and Cohen's d, provide analysis of values, and generate KDE charts of all values. -pca_plot.py - Generate PCA plot of top values from analysis with vector arrows showing value influence on plotting.

bots: -agentic_selenium.py - Selenium-based headful agentic bot running pre-determined actions with variable timing. -browseruse_llm.py - BrowserUse backend using Gemini LLM to navigate the site with human-like instructions. -crawler.py - BeautifulSoup crawler. -IWR.ps - PowerShell Invoke-WebRequest crawler. -openai_llm.py - BrowserUse backend using OpenAI LLM to navigate the site with human-like instructions. -playwright-agentic.py - Playwright-based headful agentic bot running pre-determined actions with variable timing. -playwright-headless.py - Playwright-based headless agentic bot running pre-determined actions with variable timing. -puppeteer_headless.js - Puppeteer-based headless agentic bot running pre-determined actions with variable timing. -wget_crawl.sh - Basic wget shell script to crawl the page.

All Python packages used for capture, extraction, and analysis stored in requirements.txt.

Please contact William Hatfield at william.e.hatfield@gmail.com with any questions.

The right sidebar shows repository statistics:

- Releases:** No releases published
- Contributors:** 1 (whatfield-sti)
- Languages:**
 - Python 68.3%
 - HTML 18.4%
 - CSS 5%
 - PostScript 3.3%
 - JavaScript 3.2%
 - Shell 1.8%

Figure C1: GitHub repo README screenshot