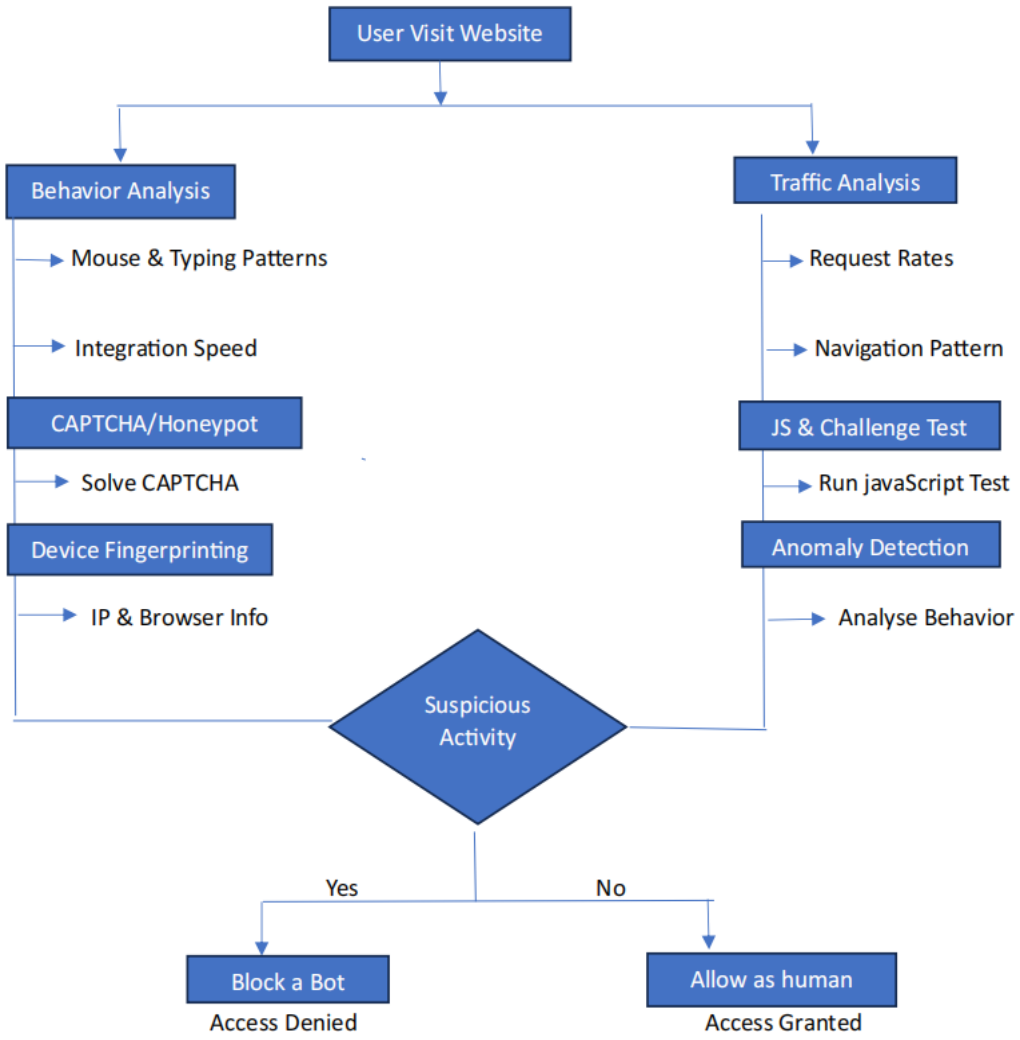
Task: Detecting Automated Traffic

Submitted to: Web3Task Pvt. Ltd

Bots automated traffic may affect the security of the websites, their performance and accuracy of data. Although some bots prove to be useful, some malicious ones can scrape, overload systems, or issue attempted unauthorized access. Detection and control of bot traffic is the key to the security of modern web applications. This report identifies the main signs of bot activity, their detection methods, and defense methods.



1. Indicators of Bot Activity

The behavioral and technical patterns of interaction in bot are different than those of human interaction:

- Artificial Movement of the Mouse: - Straight or perfectly even lines of the movement of the mouse. The human beings normally walk in slight deviations and irregularities.
- Uncharacteristic Request Frequency: - Bots are also known to produce abnormally large request rates (e.g.- 100+ requests/second with a single IP), whereas human users tend to be slower (1-5 requests/minute).

Example:- When API calls to a non-public end points suddenly upsurge after there is no activity to it, it is possible to assume that a bot is active.

- Instant Form Filling:- Forms that are typed faster than normal or fill in perfectly (e.g:- all fields filled within milliseconds).
- Input Behavioral Anomalies :- Bots post forms excessively fast (e.g. 0.1 sec response time between page load and submission) or post duplicate credential combinations. Input pauses or corrections are common among human beings.
- Abnormal Click Pattern:- Clicking items at exactly the same location each time or at very very fast rates.
- Quick Page Flipping:-Going through several pages in an unrealistically short period.
- Session Duration and Navigation patterns:-Very short sessions (e.g under 2 sec) with strict, cycle like navigation patterns that are typical of automated crawling.
- Technical Artifacts:- Bots may have suspicious user agents (e.g.: Bot/1.0) or missing necessary HTTP headers (e.g.: no Accept-language) or display geolocation inconsistency (an IP in Germany with a Japanese browser language).

2. Detection Methods

- Device and Browser Fingerprinting:- Bots can possess generic or random fingerprints. Headless browsers can be missing either plug-ins, fonts, or WebGL properties, or navigator properties can include automation.
- Honeypot trap: - Bots can be detected by introducing elements that real human users would not access, but which would be accessed by automated bots eg an invisible form field, a bogus link that the user will not see.
- Machine Learning Anomaly Detection:- Use the past traffic data to train the models to detect anomalies like an abrupt increase in 404 error. Such algorithms as Isolation Forests or LSTM networks identify small patterns not identified by rule systems.
- Rate Limiting and Adaptive Thresholds:- Use dynamic requests limit per IP address(e.g:- 50 requests per minute) since bots can avoid causing disruption to a legitimate activity. At the infrastructure level, this can be

facilitated by such tools as Nginx or Cloudflare.

3. Prevention & Mitigation

Mitigation must prioritize user experience while blocking bots.

- Progressive Challenges:- only escalate verification in case it is identified as a bot-like. Begin rate limiting then initiate CAPTCHA. Blanket CAPTCHA will be frustrating to avoid.
- Bot Whitelisting/ Blacklisting:- Allowed bots e.g.: Googlebot is reverse DNS checks and blocked IP e.g.: AllienVault OTX. Allow WAF rules to filter the bad traffic at the edge (e.g:- Bot Management by Cloudeflare).
- Apply Centric Optimization:- Identify returning users by use of cookies to overcome difficulties after first verification. As an illustration, a returning user with an authentic session token ought to evade CAPTCHA.

4. Engineering Approach and Decision Rationale

- On the engineering side, one needs to be able to scale, make it efficient, and user-friendly in terms of detection. I would introduce a layered pipeline, with cost effective checks that include request rate checking and header validation done initially, in order to support high volumes of traffic without adding high overhead costs. Suspicious sessions would be then analyzed behaviorally and they would be good signal, though they would need more processing.
- Instead of blocking traffic at once, the progressive challenges would minimize false positive as well as preserve usability. It would only be stricter in enforcing on repeated violations like rate limiting or blocking. This is a trade model of decisions on infrastructure, detection and user experience-critical factors in the implementation of solutions in actual production settings.