



Detection and analysis against shopping bots on e-commerce websites

Capstone Project Report

Team Members:

Akshay Kaikottil - akaikot1

Shivam Negi - snegi3

Ninad Shetty - nshetty1

Faculty Advisor:

Dr. Yinzhi Cao

December 6th, 2022

Table of Contents

1. Abstract	1
2. Ethics	1
3. Introduction	1
3.1. Problem Statement	2
3.2. Purpose of Study	2
3.3. Significance of Research	3
4. Literature Review & Problem Definition	4
4.1. Types of Bots	4
4.1.1. Scraping Bots	4
4.1.2. Shopping Bots or Scalping Bots	4
4.1.3. Account Takeover Bots	4
4.1.4. DDOS Bots	5
4.1.5. AIO Bots	6
4.2. Detection Methods	6
4.2.1. Browser Fingerprinting	6
4.2.2. Web Application Firewall	7
4.2.3. Honeypots	8
4.2.4. Challenge Requests	9
4.2.5. Private Access Token	10
5. Technical Solution, Design and Analysis	12
5.1. Methods Considered	12
5.2. Methods Selected	13
6. Experimentation, Evaluation and Result Analysis	14
6.1. Experimental Website	14
6.1.1. Design	14
6.1.2. Detection techniques used	15
6.2. Experimental BOT	18
6.2.1. Design	18
6.2.2. Techniques used	18
6.3. Evaluation	18
6.4. Result Analysis	18
7. Conclusion	18
8. References	19
9. Appendices	21



1. Abstract

Shopping bots may also be able to make purchases on behalf of users, either by directly interacting with online stores or by using a predefined set of rules to determine when and where to buy a particular product. These bots may be used for a variety of different purposes, including finding the lowest prices for products, tracking price changes over time, and identifying price trends.

The aim of our research and this project is to provide insight into the various ways to differentiate bots from legitimate users and the techniques used by adversaries to bypass these methods. This will provide a deeper understanding of the functionality of captchas, residential proxies, data munging, private authorization tokens and others.


We will be using multiple different methodologies like CAPTCHA, Browser Fingerprinting, IP tracking, Honeypot, Response time taken, User Agent, Private Access Tokens to create a rating model which can potentially be used to reliably predict if the visitor to the website is human user or a malicious bot.

2. Ethics

The ethics of shopping bots depends on how they are used and who is using them. Shopping bots can be a useful tool for consumers, as they can save users time and money, and can also help them to make more informed purchasing decisions by providing them with detailed information on prices and availability.

However, there are potential ethical concerns surrounding the use of shopping bots. For example, some critics argue that shopping bots can give an unfair advantage to certain users, such as professional shoppers or large corporations, who may be able to afford and use more advanced shopping bots than individual consumers. This could result in smaller, independent retailers being outcompeted by larger, more well-resourced companies.

In addition, there are concerns that shopping bots may be used to engage in unethical or illegal activities, such as price fixing, market manipulation, or fraud. For example, if a group of individuals or companies were to use shopping bots to collude and fix prices, this could result in consumers being charged higher prices than they would otherwise be able to find on their own.



Overall, while shopping bots can be a helpful tool for consumers, it is important to ensure that they are used in an ethical and responsible manner and that they do not give any individual or group an unfair advantage over others.

3. Introduction

Bots are applications that help users perform tasks for them; bots can perform tasks at scale and are quicker than humans. They provide a wide range of capabilities for people to automate routine tasks and reduce human effort. However, these bots in the hands of adversaries have recently become a nuisance to online retailers. One of these bots is known as a shopping bot.


Shopping bots are software applications that assist consumers with online shopping by searching for, identifying, and comparing products offered by online retailers. These bots have evolved into sophisticated, next-generation AI built by armies of bot builders looking to profit from the resale market. This rampage of bots has created a deficit in product availability. It denies the product to legitimate consumers, which can be considered a denial-of-service to hedge demand and raise prices, creating an unfair market [2][5].

3.1. Problem Statement

- Bots scrape e-commerce websites to gather information about products on sale and automatically make large purchases.
- There are registered companies and open-source projects creating these bots which makes them highly disguised and difficult to detect.
- These bots lead to poor customer experience as during most exclusive sales events no legitimate customers are able to make purchases. Oftentimes these bots also lead to DOSing a website.

3.2. Purpose of Study

Bots in today's world have a wide variety of use cases. Often they help us simplify our work and are widely used. They are used to supplement the number of resources we have.



In our study, we focus on the effect of bots on e-commerce websites. With the recent supply chain constraints, a vast majority of the population has not been able to purchase stuff online because of a stock deficit. It can later be noticed that these very same products that were out of stock sold for a marked-up price on a different website like eBay. These products are bought in bulk from e-commerce retailers using shopping bots and later sold for a profit. The purpose of our study is to analyze and tackle such bots [1].

Many online retailers want to block shopping bots for a variety of reasons. One of the main reasons is to protect the integrity of their pricing system and prevent unauthorized access to their website.


Shopping bots work by automatically crawling the web and collecting information on prices from a range of online stores. However, because shopping bots access websites automatically and in large numbers, they can place a significant amount of strain on a website's servers. This can slow down the website and make it difficult for other users to access it.

In addition, shopping bots may be used to engage in unethical or illegal activities, such as price fixing, market manipulation, or fraud. For example, if a group of individuals or companies were to use shopping bots to collude and fix prices, this could result in consumers being charged higher prices than they would otherwise be able to find on their own. A lot of online retailers may block shopping bots to prevent this type of behavior.

Overall, Online Retailers may block shopping bots to protect the integrity of their pricing system, prevent unauthorized access to their website, and prevent the use of bots for unethical or illegal activities.

3.3. Significance of Research

The growing industry of Bots as a Service is explored which provides a view of the premium bots which boast an overall profit of over \$2 million for the users per month. These bots come with professional UIs, 24x7 customer service, technical support teams, and CLI as well as GUI support for their products. The paper will look at different techniques used by these bots to bypass detection and the extensive community effort to shop exclusive sales of products.



Blocking shopping bots can have a number of significant effects. For online retailers, blocking shopping bots can help to protect the integrity of their pricing system and prevent unauthorized access to their websites. This can help to ensure that prices are fair and accurate and that the website is available and accessible to all users.

Blocking shopping bots can also help to prevent the use of bots for unethical or illegal activities, such as price fixing, market manipulation, or fraud. This can help to protect consumers from being charged higher prices than they would otherwise be able to find on their own and can help to maintain a fair and competitive market.

For users of shopping bots, blocking may make it more difficult to access the information and services provided by the bots. This could make it more time-consuming and difficult for users to compare prices and find the best deals, potentially reducing the effectiveness and utility of shopping bots.

Overall, the significance of blocking shopping bots will depend on the perspective of the individual or organization involved. For online retailers, blocking shopping bots can help to protect the integrity of their pricing system and prevent unauthorized access to their websites. For users of shopping bots, blocking may make it more difficult to access the information and services provided by the bots [9] [10] [14].


4. Literature Review & Problem Definition

4.1. Types of Bots

4.1.1. Scraping Bots

Web scraping is widely used today to extract useful data from webpages. In our context, it can be the availability of a certain product at a particular zip code, its price etc. Simple scraping bots are designed to crawl through websites to collect this information [3]. Bots place repeated HTTP GET requests to the target site and copy contents that come through the response packets. This process is repeated for all the routes in the target website until all the contents present are retrieved. These kinds of bots are unable to proceed when there is an HTTP form in place. Advanced bots make use of javascript to complete HTTP forms and download content normally inaccessible to simpler bots [13].

4.1.2. Shopping Bots or Scalping Bots



Scalping bots or Shopping bots are bots that automate the process of shopping online. They can add items to cart and complete the checkout without human intervention. These bots are commonly used to purchase limited edition items or items that are scarce. Malicious users often use these bots to purchase items that have a supply deficit and resell them at a premium price in order to make large profits. This results in legitimate users not being able to get these items at the MSRP price and having to pay a hefty price for them [7].

4.1.3. Account Takeover Bots

As the name suggests, account takeover bots are malicious bots that steal a person's identity. This results in malicious users gaining unauthorized access to sensitive personal or corporate data. Account users end up losing access to their accounts and fraudulent transactions are being made which can cause significant damage to both the user and the company.

Account takeover bots, also known as credential stuffing bots, are automated tools that are used by attackers to gain unauthorized access to online accounts. These bots work by using large lists of stolen username and password combinations to try and login into a variety of different online accounts. If a login attempt is successful, the bot will gain access to the account and can then use it to perform a variety of malicious activities.

Account takeover bots can be particularly damaging, as they can allow attackers to gain access to sensitive personal and financial information, and can also be used to disrupt online services and damage an organization's reputation [3]. For example, if an attacker gains access to a person's email account, they may be able to read and steal sensitive information, such as credit card numbers or login credentials for other accounts. They may also be able to send spam emails or phishing attacks to the person's contacts, potentially tricking other people into giving away their personal information as well.

To protect against account takeover bots, it is important for individuals and organizations to use strong, unique passwords for each of their online accounts, and to regularly update and change these passwords to prevent them from being stolen. It is also important to use two-factor authentication whenever possible, as this can help to prevent unauthorized access to accounts even if a username and password are stolen. In addition, it is important to be vigilant and watch for signs of account takeover, such as unusual login attempts or changes to account settings, and to report any suspicious activity to the relevant authorities.



4.1.4. DDOS Bots

DDOS stands for Distributed Denial of Service. DDOS bots are used to interrupt a website from providing service to its customers. This causes huge financial stress and can damage the company's reputation. These kinds of bots are often used in a group and the group is collectively called a botnet controlled by a single bot called the bot herder. The bot herder allows the attacker to change attack vectors when the victim adapts to previous measures. With the rise of IOT devices that have weak security, it has become extremely easy for attackers to infect these devices and in turn use them as attack bots.

DDOS bots, also known as distributed denial of service bots, are automated tools that are used to launch distributed denial of service (DDOS) attacks. DDOS attacks are a type of cyber attack in which a large number of computers, often compromised by malware, are used to flood a website or network with traffic, overwhelming the server and making it inaccessible to legitimate users.

DDOS bots are typically used by attackers to automate the process of launching a DDOS attack. The attacker will first infect a large number of computers with malware, which will then be used to control the infected machines and direct them to launch the attack. The DDOS bot will coordinate the actions of the infected computers, directing them to send a large amount of traffic to the target website or network. This can cause the website or network to crash, rendering it inaccessible to legitimate users.

DDOS bots can be particularly damaging, as they can be used to launch large-scale attacks that can take down even well-protected websites or networks. This can have a significant impact on businesses and organizations, as it can disrupt their operations and damage their reputation. For example, if a popular website is taken down by a DDOS attack, it may lose traffic and revenue, and may also suffer damage to its reputation.

To protect against DDOS attacks, it is important for individuals and organizations to use strong, up-to-date security measures, such as firewalls and intrusion detection systems, and to monitor their networks for signs of attack. It is also important to be aware of the signs of a DDOS attack, such as slow network performance or difficulty accessing a website, and to take appropriate action if an attack is detected. In addition, it is important to regularly update and patch



software and systems to reduce the risk of infection by DDOS bots and other malware.

4.1.5. AIO Bots

AIO Bots or All-In-One bots are bots that do all the things a human would do. These bots are notoriously famous among resellers due to their simple user interface and ease of use. These bots are often offered on a subscription model and are relatively easy to rent.

AIO bots, also known as all-in-one bots or sneaker bots, are automated tools that are used to purchase the limited edition or hard-to-find items, such as sneakers or collectibles, from online retailers. These bots work by automatically adding items to a user's shopping cart and completing the checkout process, allowing users to purchase items quickly and efficiently, often before they are sold out.

AIO bots are typically used by individuals and organizations to purchase the limited edition or hard-to-find items that are in high demand. Because these items are often sold out quickly, users may not be able to purchase them manually, and may instead use AIO bots to automate the process.

However, the use of AIO bots is controversial, as some critics argue that it gives users an unfair advantage over other shoppers, and can also result in inflated prices for limited edition items. In response, some online retailers have implemented measures to detect and block AIO bots, and may also cancel orders that are believed to have been placed using a bot.

Overall, AIO bots are a type of automated tool that can be used to purchase the limited edition or hard-to-find items from online retailers. While they can be useful for some users, their use is controversial, and some online retailers have taken steps to block or restrict their use [6].



4.2. Detection Methods

4.2.1. Browser Fingerprinting

Browser fingerprinting is a tracking technique that accumulates enough information to identify a single user over multiple browser sessions, even when the person is browsing incognito or using a VPN.


Because a user may quickly clear their cookies or use an ad blocker that blocks cookies, browser fingerprinting is considered a superior tracking tool to cookies. Even if there is no cookie match, one can utilize browser fingerprinting to identify who is visiting their website. Browser fingerprinting is effective because it can identify and monitor any request that exhibits suspicious behavior across browsing sessions. Malicious actors cannot hide behind a VPN or a different browser. They would require a completely distinct apparatus for each request, which would be prohibitively expensive.

There are various different fingerprinting techniques like HTTP, and TLS. TCP, Mobile, Canvas, WebGL, Media device, audio, etc. However, Bot operators work in collaborative groups. Some of these communities sell toolkits or even full digital fingerprints that have been stolen. Digital fingerprints are essentially clones of actual user sessions/ browser data that could potentially be fed into bot frameworks to nearly identically impersonate a human user using a browser. This could enable them to deceive both mitigation vendors' data collection and categorization processes [12].

4.2.2. Web Application Firewall

The WAF examines incoming traffic for GET and POST-based HTTP requests and employs a set of predefined rules to filter out suspicious traffic with known attack signatures. Although not a very strong technique to block or detect bots, it is still a critical piece of technology that should be deployed.

WAFs are generally IP address centric and can block traffic from known malicious user agents, and IP addresses, including countries entirely. AbuseIPDB and Project Honey Pot are two notable community-driven projects that collect data on malicious or unwanted IP-related activity. They collect data about malicious or unwanted Internet traffic and process it to generate actionable summaries about IP addresses [3].



However blocking just ranges of IP addresses is not entirely effective, they force bot makers to employ expensive counter technologies using IPv6, botnets, and IoT deployments. Also, bots generally look for flaws in business logic rather than software vulnerabilities. Bots are now widely distributed. Bot programmers are becoming more astute, purposefully building their bots to avoid standard WAF bot detection solutions, and their techniques are constantly evolving.

One of the main advantages of a WAF is that it provides an additional layer of security for web applications. By blocking malicious traffic, a WAF can help to prevent attackers from accessing sensitive data or disrupting the operation of a web application. This can help to protect the web application, its users, and the data it stores from a variety of different types of attacks.

Another advantage of a WAF is that it can be easily configured and customized to meet the specific needs of a web application. WAFs often come with a range of pre-configured security rules and settings, which can be easily customized and tailored to the specific requirements of a web application. This can make it easier for organizations to protect their web applications without having to develop their own security solutions from scratch.

However, there are also some disadvantages to using a WAF. One of the main concerns is that WAFs can be difficult to configure and manage correctly. Because WAFs operate by inspecting and blocking incoming traffic, they can potentially block legitimate traffic as well as malicious traffic. This can result in false positives, where legitimate users are unable to access a web application, or false negatives, where malicious traffic is able to bypass the WAF and access the web application.

Another disadvantage of WAFs is that they may not be effective against all types of attacks. While WAFs are designed to protect against a wide range of attacks, they may not be able to protect against more advanced or sophisticated attacks, such as zero-day exploits or targeted attacks. This can leave a web application vulnerable to attack, despite the presence of a WAF.

Overall, the advantages and disadvantages of a web application firewall depend on the perspective of the individual or organization involved. While a WAF can provide an additional layer of security for web applications, it can also be difficult to configure and manage correctly, and may not be effective against all types of attacks.



4.2.3. Honeypots

As the name implies, a honeypot is a trap meant to trick bots and computer programs into unwittingly revealing their identities. The goal is to supply something that will attract the bot while remaining unseen or concealed from legitimate human users.

They trap malicious actors by designing appealing targets filled with flaws. They provide for threat identification ahead of future attacks. Trap setters can learn about the types of attacks bot cyberattackers might use by getting hackers and bots to latch onto a bogus target. They make excellent intrusion detection systems. They help us understand these bot cyber attackers better, which is valuable feedback while designing security features for the websites [4].

One major advantage is that a honeypot can help to deflect potential threats away from the rest of the network. Because the honeypot is isolated, any malicious activity on it will not be able to spread to other parts of the network. This can help to protect critical systems and data from being compromised.


However, there are also some disadvantages to using a honeypot. One is that it can be resource intensive to set up and maintain a honeypot, as it requires dedicated hardware and software, as well as ongoing monitoring and analysis. This can be costly and may not be feasible for all organizations.

Another disadvantage is that a honeypot can also be a target for legal action. Because a honeypot is essentially a trap, some attackers may feel that they have been entrapped and may try to take legal action against the organization that set it up. This can create additional risks and liabilities for the organization.

Overall, the effectiveness of a honeypot as a security tool depends on the specific circumstances and how it is implemented. It can be a valuable tool for detecting and deflecting threats, but it is not a replacement for other forms of security and should be used carefully and in conjunction with other security measures.

4.2.4. Challenge Requests

Challenge response tests are used to determine if the user is a bot or a legitimate user. They are popularly known as CAPTCHAs and everyone would have seen them once in a while. The right CAPTCHA needs a few key things:

- 
- High accuracy in identifying bots and should have the accurate amount of difficulty to identify bots.
 - No hindrance to user experience and should be fast.
 - Compliance with global and local data privacy regulations.

However, the effectiveness of CAPTCHA tests in stopping bots can vary. Some bots are designed specifically to bypass CAPTCHA tests, using advanced image recognition algorithms or other techniques to interpret distorted images or other challenges presented by the test. In these cases, CAPTCHA tests may not be effective at preventing bots from gaining access.

On the other hand, well-designed and properly implemented CAPTCHA tests can still be effective at stopping many bots. These tests can help to prevent spam, fraud, and other security threats by blocking automated software from accessing a system.

One disadvantage is that CAPTCHA tests can be frustrating for some users. These tests often require the user to interpret distorted images of letters and numbers, or to complete other challenges that may be difficult to understand. This can cause delays and inconvenience for legitimate users, and may even cause some users to abandon the site or system altogether.

Another disadvantage is that CAPTCHA tests can be expensive and resource-intensive to implement and maintain. This can be particularly challenging for small or low-budget organizations that may not have the resources to devote to implementing and maintaining a CAPTCHA system.

Additionally, some CAPTCHA tests can be difficult for users with visual impairments or other disabilities to complete. This can create barriers to access for these users and may violate accessibility laws in some cases.

Overall, while CAPTCHA tests can be an effective security measure, they also have some disadvantages that organizations should consider before implementing them. CAPTCHA tests may not be able to stop all bots, but they can still be an effective security measure when used as part of a broader security strategy.

4.2.5. Private Access Token

PAT or Privacy Access Tokens is a new technology introduced by Apple in their latest platform offering. It is built around Internet Engineering Task Force's (IETF)'s Privacy Pass Protocol. It is an application layer mechanism that revolves around the creation of a token by a server and its anonymous redemption by various clients. The main idea behind PAT is privacy, ensuring that users are not being tracked by clients in the name of user verification.

A user's access to a specific resource or service is authenticated using a private access token (PAT), which is a one-of-a-kind string of characters. Usually, an API or other service that demands authentication to access particular features or data is used in conjunction with this kind of token.

Private access tokens are frequently employed to grant secure access to sensitive data or resources, such as financial information or exclusive software [15]. They are generally produced by the service or resource provider, and only authorized users are given access to them. This enables the provider to keep tabs on consumption as well as manage who has access to their resources.

Data Exchange Flow:

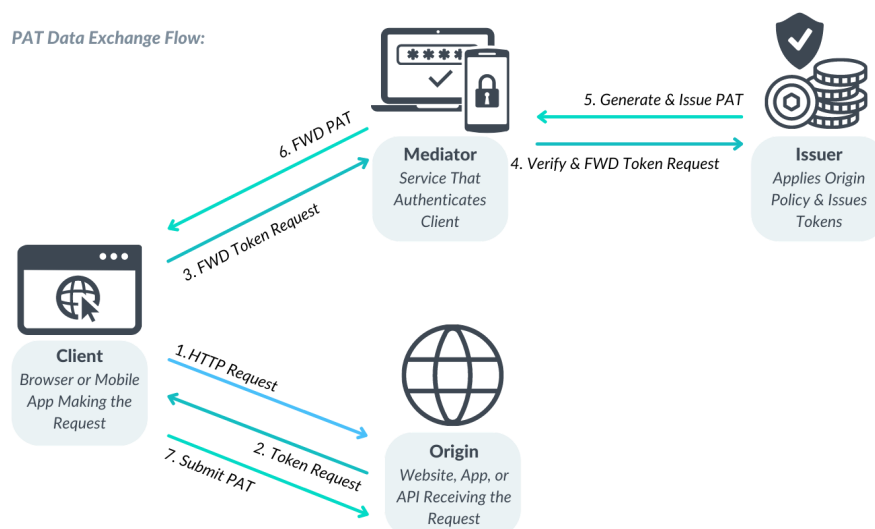



Fig 1: Private Access Token (PAT) Data Flow



With PAT verification, first the client(browser/mobile app) submits a request to the origin, the origin being any website/application/API. The origin then redirects the client to the issuer to obtain a token for verification. The issuer generates the token for clients on behalf of the origin via a mediator. The mediator is a very important part of this system and is responsible for anonymously fingerprinting the client and passing this information on to the issuer. Once the issuer is satisfied with the request and the origin's policies, it finally issues the tokens to clients once again through the mediator. Now the client can submit the PAT back to the origin and establish an identity as a non-bot user.

PAT is a huge win over traditional captcha since the verification provider is no longer able to track the user's activity.

Common steps in the process of using private access tokens for authentication might include:

- The user requests access to a protected system or service.
- The system or service generates a private access token and provides it to the user.
- The user presents the private access token to the system or service as proof of their identity.
- The system or service verifies the authenticity of the private access token.
- If the private access token is valid, the system or service grants the user access to the protected resources.

The PAT protocol is a recent development. It is still in a form that reflects a lot like public beta, and could potentially take time for the implementation to become more refined. Attackers are skilled at identifying and exploiting weaknesses in new products, which may result in an increase in security incidents until the technology has matured.

The protocol recommends using rate limiting, tying the token to the IP subnet, and limiting the validity of the token as a security feature. However, these are well-known security measures that attackers have learned to bypass in the past, and may not be sufficient to prevent abuse of the tokens. This is why there is a need to collect a variety of data to accurately evaluate traffic and protect against attacks rather than rely on a generic filtering process.

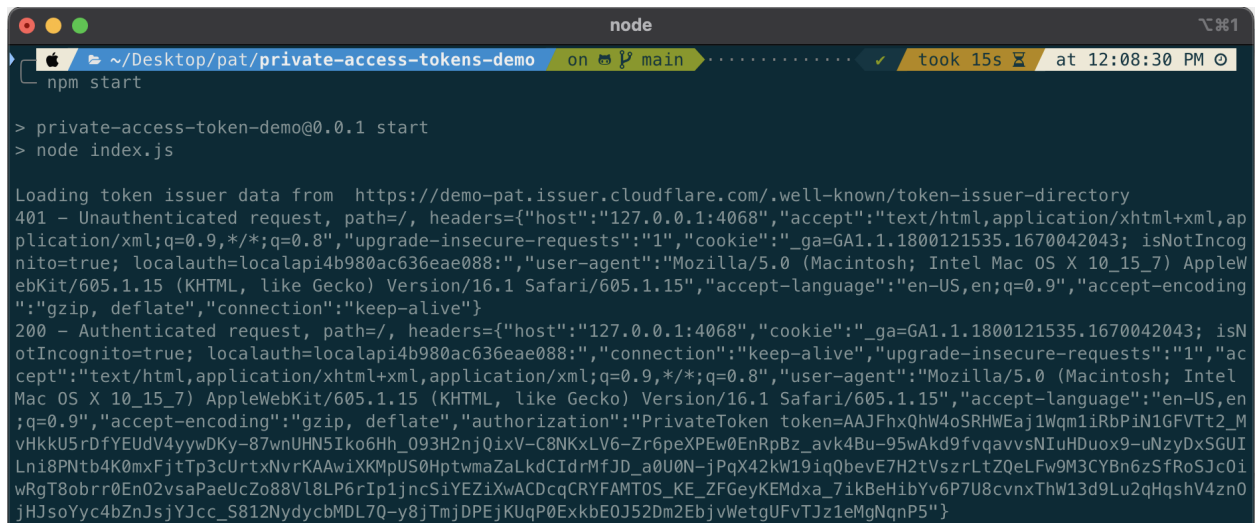
5. Technical Solution, Design and Analysis

5.1. Methods Considered

Considering the various techniques available to us, we conducted an extensive literature review. We considered the below-mentioned techniques to be implemented on our test website.

- Captcha
- Browser fingerprinting
- IP tracking
- Honeypot
- Response Time taken
- User Agent
- Private Access Tokens

We considered and tested out an implementation for private access tokens in a test environment. We were able to successfully obtain a private access token and successfully authenticated the token [15].



```
node
~/Desktop/pat/private-access-tokens-demo on main
npm start

> private-access-token-demo@0.0.1 start
> node index.js

Loading token issuer data from https://demo-pat.issuer.cloudflare.com/.well-known/token-issuer-directory
401 - Unauthenticated request, path=/, headers={"host":"127.0.0.1:4068","accept":"text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8","upgrade-insecure-requests":"1","cookie":"_ga=GA1.1.1800121535.1670042043; isNotIncognito=true; localauth=localapi4b980ac636eae088","user-agent":"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.1 Safari/605.1.15","accept-language":"en-US,en;q=0.9","accept-encoding":"gzip, deflate","connection":"keep-alive"}
200 - Authenticated request, path=/, headers={"host":"127.0.0.1:4068","cookie":"_ga=GA1.1.1800121535.1670042043; isNotIncognito=true; localauth=localapi4b980ac636eae088","connection":"keep-alive","upgrade-insecure-requests":"1","accept":"text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8","user-agent":"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/16.1 Safari/605.1.15","accept-language":"en-US,en;q=0.9","accept-encoding":"gzip, deflate","authorization":"PrivateToken token=AAJFhxQhW4oSRHWEaj1Wqm1iRbPiN1GFVtT2_MvHkkU5rDfYEUdV4yywDKy-87wnUHN5Iko6Hh_093H2njQixV-C8NKxLV6-Zr6peXPEw0EnRpBz_avk4Bu-95wAkd9fvqavvsNIuHDuox9-uNzyDxSGUILnI8Pntb4K0mxFjtTp3cUrtxNvrKAAwiXKmpUS0HptwmaZaLkdCidrMfJD_a0U0N-jPqX42kW19iqQbevE7H2tVsZrLtZQeLfw9M3CYBn6zSfRoSjC0iWRgT8obrr0En02vsaPaeUcZo88Vl8LP6rIp1jncSiYEZiXwACDcqCRYFAMTOS_KE_ZFGeyKEMdxa_7ikBeHibYv6P7U8cvnxThW13d9Lu2qHqshV4zn0jHJsoYyc4bZnJsYJcc_S812NydyCbMDL7Q-y8jTmjDPEjKUqP0ExkbE0J52Dm2EbjvWetgUFvTJz1eMgNqnP5"}
```

Fig 2: Private Access Token (PAT) - Successful token generation



Fig 3: Private Access Token (PAT) - Successful user authentication

Considering that a change in PAT architecture can not be implemented within the scope of this project, we decided not to include its implementation as part of our solution. Future work would include modification of the PAT architecture to include browser fingerprint data to keep track of verified users. This will enhance security as the current implementation only makes use of basic security features for detection and verification.

PAT currently makes use of information such as its IP address, account name, or device identifier. Anonymizes a Client to an Issuer and relays information between an anonymized Client and an Issuer. Also, The protocol suggests using rate limiting, tying the token to the IP subnet, and limiting the validity of the token to detect abuses. These unfortunately are known security measures that attackers learned to circumvent years ago and may not be enough to prevent abuse with the tokens. This could be overcome by making use of browser fingerprinting in conjunction with PAT.

5.2. Methods Selected

We selected the below methods based on their accuracy and the feasibility of their implementation for this project.

- Captcha
- Browser fingerprinting
- IP tracking
- Honeypot
- Response Time taken
- User Agent



We have used these techniques in tandem to create a rating model which can be used to predict if the visitor to the website is a human or a malicious bot.

6. Experimentation, Evaluation and Result Analysis

We set up a test environment on Python and used flask as the backend for our experimentations. Along with this, an auto-checkout bot is developed in selenium which is used to auto-checkout products from the experimental website.

6.1. Experimental Website

6.1.1. Design

The website is developed to recreate the simple functionality of a shopping website. The initial landing page contains a login button that redirects to our login page which is protected behind a captcha check page.

The landing page greets the user with a bot image and a 'Bot Attack' title on the top. This has been designed using the Bootstrap 5.1 framework. This provides it a Sass for modular and customizable architecture. The navbar has been designed with only the necessary details in mind. This is why only a 'login' button is visible.

For a more accurate implementation of an e-commerce website like Amazon or Flipkart, there would be more options along with the login page. Providing different shopping deals for the user to use.

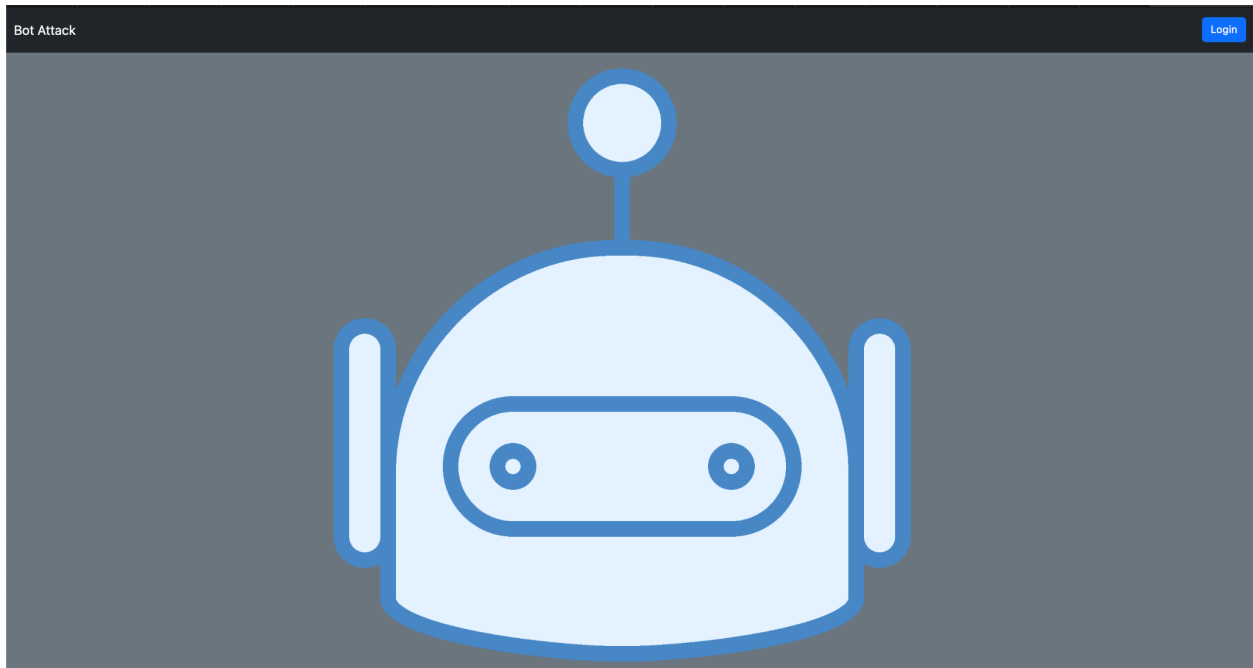


Fig 4: Experimental website - Landing page

The login page provides a login functionality to keep track of the user. We have set up default test accounts for experimental purposes. The credentials are provided in the readme.md file.

This page also includes a hidden honeypage functionality which provides a parameter for the bot detection engine to use for the identification of the bot. Honeypage is a 'trap' functionality that is hidden from a legitimate user. A custom checkbox with a hidden input field is added to the DOM of the login page. The checkbox has a name for 'honeypot' and can be enabled by a bot randomly surfing through the website and trying to go over all the possible flows in the website.

Fig 5: Experimental website - Login page

Upon login, a checkout page is provided to place an order. The functionality of this page is limited in scope to only what is essential for the effective testing of the project. The page presents a sample item on the e-commerce website called the 'Apple Watch Design' this item is available for the user to buy for \$185. After adding all the details of the user like address, payment details, and bank details there is a button that provides an option to complete the payment.

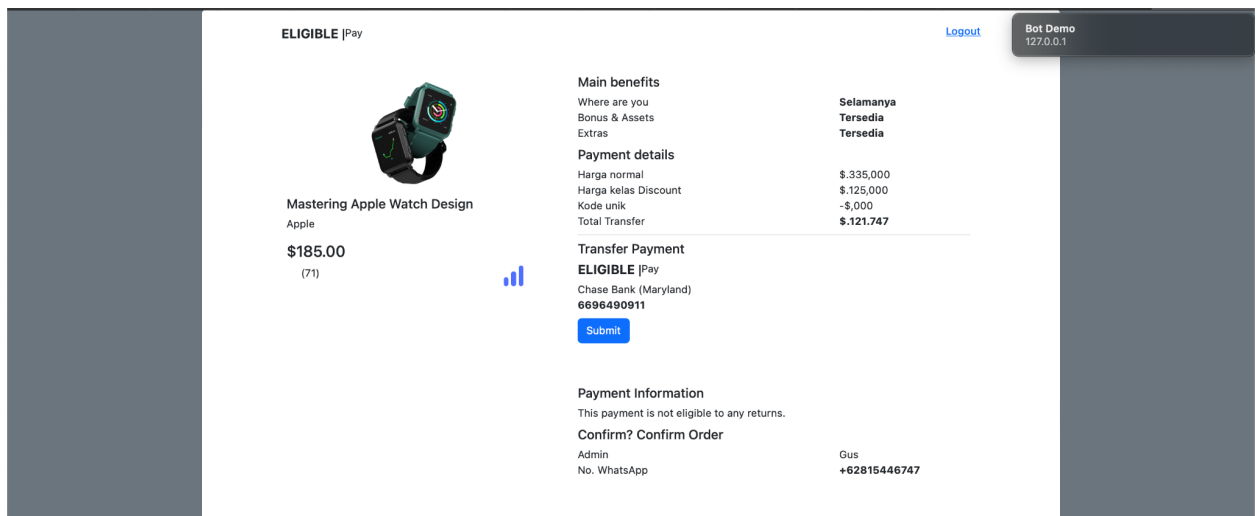


Fig 6: Experimental website - Checkout page

6.1.2. Detection techniques used

We have implemented all the methods selected in the previous section.

- Captcha

We have implemented google ReCaptcha on our website. Clicking on the login page will redirect a user to a captcha verification page, which has to be completed in order for the visitor to proceed to the login page.

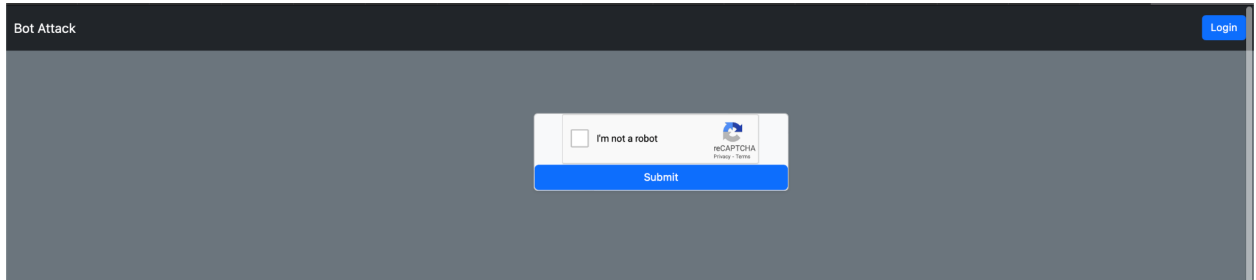


Fig 7: Experimental website - Captcha page

The result of the captcha verification is checked with google through the flask backend with the use of the `is_human()` function. The function calls google's API to verify the validity of the captcha token issues. It is valid for 2 mins before the user needs to retake the captcha challenge if his previous challenge goes unverified. This data is then passed on to the final checkout page where our model evaluates whether the user is human or a bot.

- Browser fingerprinting

Our website makes use of fingerprintjs [17] to fingerprint our visitors' browsers. This helps us identify visitors who might make use of different user accounts to visit the website with malicious intentions. We are making use of this data to mark known bots with this technique on the final checkout page [8].

- IP Tracking

Residential IPs are more expensive than data center IPs so bot providers prefer using data center IPS. On the 209 end-of-year reviews done by data dome, out of all of the bad bot requests they found out 49.9 % were using data center IPs. Thus, on our website, we are making use of the IP of the request to add to the likelihood of a bot.

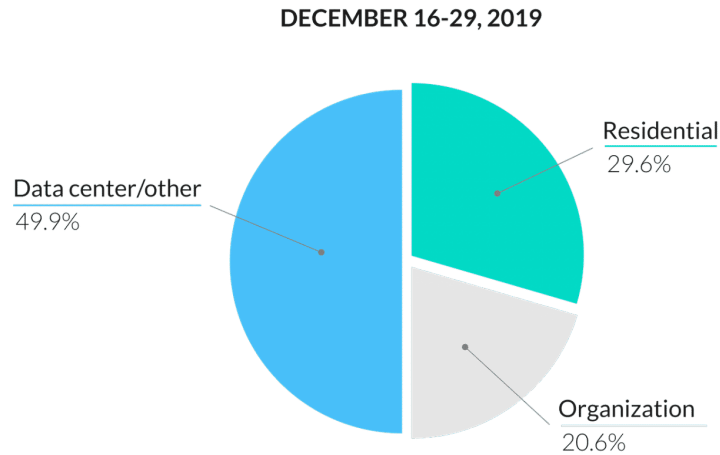


Fig 8. Bot IP statistics


- Honeypot

Honeypot is a mechanism to include an interaction on the webpage which is only accessible to bots. Our login page has a checkbox with the label login which is set to hidden. A human user will not be able to access this checkbox. When a malicious bot triggers this checkbox, we will know for certain that it was done by a bot. The value of the honeypot variable is also used to determine the authenticity of the user.

- Response time taken

Considering that an average user takes time to make logical decisions before buying a product, we make use of response time as a parameter to determine the authenticity of the visitor. We compute the time taken from the user visiting a web page to the time is taken for him to complete the checkout. This is further evaluated with other parameters collected to verify the user's authenticity.

Bots are built to move considerably more quickly than human users. Therefore, we can tell the difference between genuine human users and bots by timing how long it takes a user agent to accomplish a job (such as filling out and submitting a form). Although a bot operator can slow down a bot's operation, most bot operators want to complete tasks as soon as possible because most bots use resources that might be expensive (for example, when the bot is rented).



The objective of time measurement detection is to drastically slow down bot activity to deter bot operators. Ideally, they'll lose patience and stop focusing on your website.

- User Agent

User agent tells us the details of the browser used to visit the web page. Although this can be easily spoofed by the bot, it is a good measuring factor to determine the visitor's authenticity.

6.2. Experimental BOT

We developed a bot to provide auto-checkout functionality on the experimental website. This bot imitates the functionality of a human user by functioning as a headless browser and browser configuration. The bot can automatically login as a user on the experimental website and purchase an item, 'Apple Watch'.

6.2.1. Design


The web browser interaction is automated using Selenium. A major reason for this is to ensure as close a representation of a human user as possible. The other option of directly using API access does not provide the functionality of mouse movement which is a major factor in detecting bot behavior.

The webdriver being used in the bot is the latest Chrome driver which is being auto-configured with the selenium web driver. The bot reaches each page and then, according to the 'xpath' configured for the web page, navigates to the required inputs. For example, when the bot is trying to login the username and password is supplied by identifying the input for fields using `xpath="//input[@name='email']"`. [16] [11].

6.2.2. Techniques used

The bot spoofs its User Agent as shown below in order to avoid detection as a headless browser without user agent.

```
Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.0.0 Safari/537.36
```



To bypass the captcha the bot developers have a practice of implementing captcha farms. These are API services that automate the enlistment of humans to solve CAPTCHAs. This is directly using humans at very low pay to solve CAPTCHA challenges instead of using AI.

The bot also takes into consideration slow page loading times and ensures that the page is fully loaded before trying to fetch a particular tag from the web page to work on. This is also complemented by the fact of setting page load timeout to ensure that the bot doesn't get stuck at a particular page.


6.3. Evaluation

The bot is successful in logging into the experimental website and ordering the product with the right credentials. The auto-checkout functionality can easily place multiple orders even before legitimate users are able to place an order. It is a bot application that does not require any overhead for the adversary proving its effectiveness is hiding as a legitimate user profile.

```
Fingerprint: 064e4c440a6346bbe383c67aabe12600
Client ip: 127.0.0.1
User Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/107.0.0.0 Safari/537.36
Is this a webdriver: true
Time Elapsed: 1.0584945678710938
```

The detection methods implemented in the updated version of the website detect the experimental bot successfully. The captcha is very difficult to bypass even by industry-grade bots and requires human interaction to solve. The adversaries can still employ Captcha Farms for solving the bots but it adds an expense for the adversaries. Along with this, the captcha can change its difficulty based on the predicted likeliness of a bot. Making it increasingly difficult to solve it while keeping the legitimate user experience easy.

The honeypage input implemented on the 'login' page provides a "trap" for the bot to accidentally enable. The input field can provide additional user information to the website administrators. This can be especially helpful when the




bot unassumingly lands on the honeypage input, as the input is completely inaccessible to a normal user. The information provided by enabling this input field provides necessary information for the bot detection engine in the backend servers to predict if the user is a bot. This input is especially successful when detecting scraper bots that try to go through all of the inputs on a website in order to crawl the available data on a website.

The webdriver input field is a property that is defined according to the webdriver specs [16]. This field is being used to identify the probability of the bot to a very high degree. As this input field specifies the existence of a platform or a language-driven control of the browser DOM elements. Whenever the bot website bot detection engine detects the presence of the driver input the user is flagged as a bot.

In our results, the IP-based tracking detection method could not be extensively tested. The detection method works on the basis of identifying the categorization of the IP to either cloud-based IP or a residential IP. During a Distributed attack on a website, the adversaries are prone to use cloud-based IPs which can be easily detected by the IP tracking engine to detect and flag the existence of a bot. Stopping the usage of cloud-based IP is very crucial as these IPs tend to have a lot of advantages for an adversary as compared to residential proxies. The availability, speed, and prices of a cloud-based IP are most suitable for an adversary. This keeping a higher weightage for bot detection makes it difficult for the adversaries to perform Distributed bot attacks.

Device fingerprinting will allow the collection of more information from the user about their software and hardware device. The fingerprintJS library queries the browser attributes and computes the hash id respective to each user. This ensures that it is not easy for the attacker to change their fingerprint without taking additional steps. The computed hash works with about 60% accuracy [17] and once, is detected as a bot. The fingerprint can be used to show a different user interface to the bot users. This effectively will decrease the cost of the ad revenue and the data center workload of the website.

The response time is stored from the time of login to the time of checkout of the product from the experimental website is effective in recognizing bots. This parameter can distinguish a legitimate user from a bot user by separating the inhuman response time to a connection request. This factor also slows down the overall speed of a distributed bot attack on the website as in order to tackle this



detection method the adversaries will have to add a time delay in their input process and overall workflow.

There are multiple contributing factors to this and the weightage offered to the different parameters is key in the detection of the bot. Although the parameters added to the website are limited to six parameters that have been identified they are effective in identifying the experimental bot. Some of the parameters considered in the experimental website like the web driver, and user agent have very less false positives in the detection of bots and are very effective if not known to the adversary.

6.4. Result Analysis


The “Completely Automated Public Turing Test to Tell Computers and Humans Apart” added on the login page to provide a challenge-response test is very effective. The usage of the tests should be implemented in all websites to thwart the bots. The only real way to bypass these bots is to add human interaction to the bot process. This is why this is very effective in stopping bots.

The new tools like reCAPTCHA are more secure, and privacy-compliance and user-friendly CAPTCHA solutions are better. Depending on the user 'bot' score the reCAPTCHA reduces or increases the complexity of the test being provided to the user. Maintains user experience and reduces the overall time spent in solving CAPTCHAs by the users.

Fingerprint analysis shows that it can be a very important tool for blocking bots. Once a bot user is identified the adversary cannot hide behind a VPN, an incognito session, or a new browser. This would make the fraud operation costly forcing the adversary to switch devices and take other extensive steps to hide.

The Client IP again is an effective way to identify basic bots and keep the operating costs of bots expensive. Even though buying residential IPs is becoming cheaper day by day, blocking bots on the basis of IPs ensures that the adversary has to keep cycling through their available IP proxies.

Also about honeypots, one advantage we can note is that they can be a very effective way to detect and track attempts at unauthorized access to a system. Because a honeypot is designed to look like a legitimate part of the entire system,



attackers and most importantly bots are often drawn to it, thinking they have found a vulnerable system. This allows security teams to observe their actions and gather intelligence on their tactics and techniques. Since bots are ever-evolving, there is a greater need for defenses deployed to evolve ahead of them.

6.5. Future Work


The current experimental website gathers only six parameters of users visiting the website. To improve the accuracy of the bot detection different parameters based on the session and account information can be added to the detection. Parameters like order history, login frequency, and account creation date.

As the website is able to record more data the engine for identifying bots will improve in fingerprinting bots from legitimate users. More data points will directly help in improving the accuracy of the engine and an ML model can be put in place. With the use of machine learning algorithms and models, we can identify underlying patterns that are able to discern between human and robot behavior. Along with this techniques can be employed to not only prevent and detect the bots but also to fool the bots. This can be done by showing modified content to the bots so that it becomes difficult for the adversary to identify if the bot has been blocked or not.

Privacy is also an important issue when it comes to bot detection methods. This is because many of these methods involve collecting and analyzing data from individuals or organizations, which can potentially be sensitive or personal. If this data is not handled properly, it can lead to privacy breaches that can seriously affect the individuals or organizations involved. Therefore, bot detection methods must be designed and implemented to take privacy into account. Appropriate measures are put in place to protect the privacy of individuals and organizations. This might include, for example, using anonymized or aggregated data, or obtaining consent from individuals before collecting and analyzing their data.

7. Conclusion


Detecting and defending against shopping bots is important for ensuring the smooth and fair functioning of e-commerce websites and protecting the interests of both businesses and consumers. Through our research and project, we contribute



towards ensuring the security and integrity of websites and protecting the interests of both businesses and consumers. If we incorporate a composition of multiple different techniques targeting different attack vectors in our defense and detection strategy, we will likely have a higher success rate. While building such technologies, it is also critical that we keep preventing user privacy as a high-priority item.

8. References

- [1] L. Winkie, "The unstoppable machines behind the game console shortage," 2022. [Online]. Available: <https://www.theverge.com/2022/5/25/23137789/aio-buying-bots-ps5-xbox-series-x-console-shortage>. [Accessed: 15-Oct-2022]
- [2] G. Somanath, "Can't find that perfect gift? Blame the bots," vol. 2021, no. 3, pp. 6–8, Jan. 2021, DOI: 10.1016/S1361-3723(21)00028-2. [Online]. Available: <https://www.magonlinelibrary.com/doi/abs/10.1016/S1361-3723%2821%2900028-2>. [Accessed: 15-Oct-2022]
- [3] T. S. Reddy, "Impact of Bots on {eCommerce} and Bot Detection Methods," 2018 [Online]. Available: <https://www.usenix.org/conference/scainet18/presentation/reddy>. [Accessed: 15-Oct-2022]
- [4] B. Amin Azad, O. Starov, P. Laperdrix, and N. Nikiforakis, Web Runner 2049: Evaluating Third-Party Anti-bot Services, vol. 12223. Cham: Springer International Publishing, 2020, pp. 135–159 [Online]. Available: http://link.springer.com/10.1007/978-3-030-52683-2_7
- [5] R. Heilweil, "How Bots Bested the \$1 Billion Sneaker Resale Industry," 20-Jul-2017. [Online]. Available: <https://www.forbes.com/sites/rebeccaheilweil/2017/07/20/how-bots-bested-the-1-billion-sneaker-resale-industry/>. [Accessed: 15-Oct-2022]
- [6] PerimeterX, "What is web, price, and content scraping?" [Online]. Available: <https://www.perimeterx.com/solutions-by-threat/web-scraping/>. [Accessed: 16-Oct-2022]
- [7] J. Lapienyte, "Can't find these items? Scalper bots are to blame," 2022. [Online]. Available: <https://cybernews.com/news/bots-snatch-popular-items/>. [Accessed: 15-Oct-2022]
- [8] B. Amin Azad, O. Starov, P. Laperdrix, and N. Nikiforakis, Short Paper - Taming the Shape Shifter: Detecting Anti-fingerprinting Browsers, vol. 12223. Cham: Springer International Publishing, 2020, pp. 160–170 [Online]. Available: http://link.springer.com/10.1007/978-3-030-52683-2_8

- 
- [9] "Millions of bad bots attacks on e-commerce sites detected," New Delhi, 2020 [Online]. Available: <https://www.proquest.com/newspapers/millions-bad-bots-attacks-on-e-commerce-sites/docview/2466297227/se-2>
- [10] "E-Commerce product comparison portal for classification of customer data based on data mining," vol. 51, pp. 166–171, 2022, doi: 10.1016/j.matpr.2021.05.068. [Online]. Available: <https://www-sciencedirect-com.proxy1.library.jhu.edu/science/article/pii/S2214785321036440>. [Accessed: 15-Oct-2022]
- [11] Sanaa, M. Haris, Samadyar, and M. A. Shah, "The Price Scraping Bot Threat on E-commerce Store Using Custom XPATH Technique." The Institute of Electrical and Electronics Engineers, Inc. (IEEE), Piscataway, 2021 [Online]. Available: <https://www.proquest.com/conference-papers-proceedings/price-scraping-bot-threat-on-e-commerce-store/docview/2598260218/se-2?accountid=11752>
<http://findit.library.jhu.edu/resolve>
- [12] A. Vastel, P. Laperdrix, W. Rudametkin, and R. Rouvoy, "FP-STALKER: Tracking Browser Fingerprint Evolutions," 2018, p. 728, DOI: 10.1109/SP.2018.00008 [Online]. Available: <https://hal.inria.fr/hal-01652021>. [Accessed: 16-Oct-2022]
- [13] Cloudflare, "What is content scraping? | Web scraping." [Online]. Available: <https://www.cloudflare.com/learning/bots/what-is-content-scraping/>. [Accessed: 16-Oct-2022]
- [14] L. Bressler and M. Bressler, "Beware the evil bots: e-commerce thieves and spreaders of 'fake news,'" vol. 8, Jan. 2019.
- [15] Private access token demo [Online]. Available: <https://github.com/m-keller/private-access-tokens-demo> [Accessed: 18-Nov-2022]
- [16] "W3C Editor's Draft", 25-Oct-2022 [Online]. Available: <https://w3c.github.io/webdriver/#interface> [Accessed: 18-Nov-2022]
- [17] "Browser Fingerprint", <https://github.com/fingerprintjs/fingerprintjs> [Accessed: 18-Nov-2022]



9. Appendices

- Private access tokens demo - <https://github.com/m-keller/private-access-tokens-demo>
- Website code base - <https://github.com/kaikottil/websecwebsite>
- Bot code base - <https://github.com/ShivamNegi/Bot>