API MISUSE & ABUSE

Prepared by :

- Aishani
- Dibya
- Shreya
- Vignesh

WHAT IS API MISUSE AND ABUSE?

Large and complex softwares are developed with integrated components using application programming interfaces.(APIs). When developers use these, they often make mistakes that lead to security vulnerability, bugs, system crashes. These mistakes are called API misuses.

API abuse is the misuse of API to carry out malicious activities.
These include:

- Account takeovers
- Credential Stuffing
- Bot content scraping
- Fake account creation

ACCOUNT TAKEOVERS

Account takeover is a kind of an identity theft where a fraudster illegally gets access to a victim's bank or any other account and uses it to indulge in fraudulent transactions.

To prevent such account takeovers, time-worn practices were used such as blacklisting certain IP addresses, limiting login attempts, using CAPTCHAs, robust authentication process etc. However in the recent years, advanced API based approaches are being implemented to detect such takeovers and prevent them.

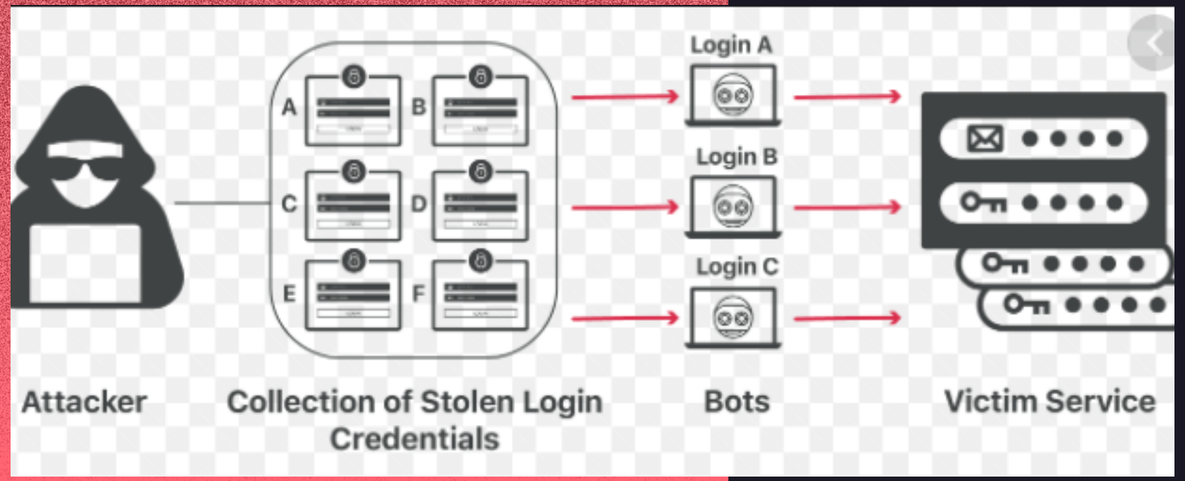
A few common ways to identify such takeovers:

- Accounts with multiple IP addresses of different countries. This information is usually available through the log data.
- Multiple accounts having details changed to a particular shared detail within a short span of time.
- Multiple accounts linked to the same device
- New account details, new device and a new delivery address all within 2-3 days.

CREDENTIAL STUFFING

Credential stuffing is a cyberattack technique in which attackers use records of compromised user credentials to breach a system. The attack employs bots for automation and is based on the assumption that many users reuse usernames and passwords across multiple services.

A few prevalent ways to detect and prevent credential stuffing are as follows:



- Multi-Factor Authentication (MFA)
- Using a CAPTCHA verification
- Device Fingerprinting (fingerprint is a combination of parameters like operating system, time zone, user agent, etc.)
- IP Blacklisting
- Rate-Limit Non-Residential Traffic Sources
- Disallow Email Addresses as User IDs
- Block Headless Browsers

FAKE ACCOUNT CREATION

The development in online networking has exposed people to various issues, including the risk of revealing false data by generating fake accounts resulting in the spread of malicious content. Fake accounts are a familiar way to forward spam, commit fraud and abuse through an online social network.

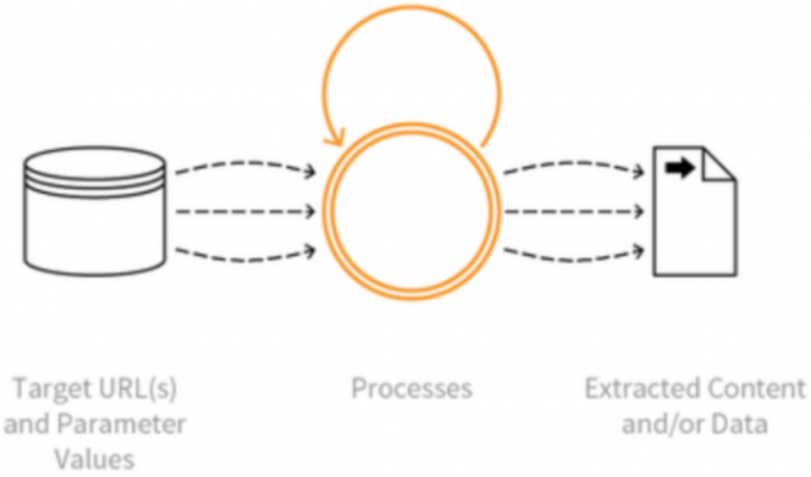
Common defenses against fake account creation attacks can include:

- Employing user behavior analysis to detect abnormal activity.
- Enforcing user verification methods to make it difficult for bots to create accounts.
- Using an automated bot protection tool to prevent fake account creation attacks.

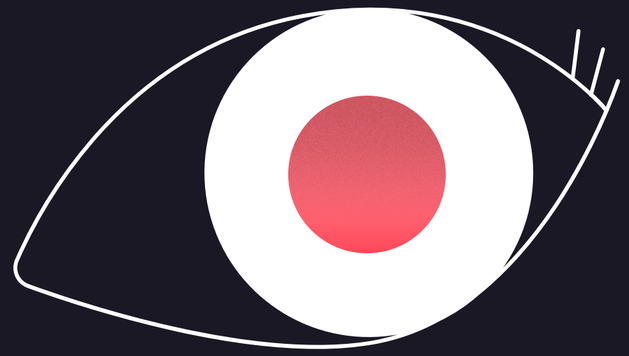
Because new account creation is a vital process for online businesses, a defense methodology must accurately identify fake account creation attempts without creating an excess of user interface friction.

BOT CONTENT SCRAPING

Web API Content Scraping is the process of extracting data or information from websites and distributing it elsewhere. This is an illegal activity done without the consent of the owner of the source. Manually scraping the content is an uphill task, and hence scrapers deploy sophisticated programs known as bots and scrape many pages illegally.
A few ways in which you can detect the presence of scraper bots in your website are :



- Monitoring new or existing user accounts with high levels of activity and no purchases
- Detecting abnormally high volumes of product views as a sign of non-human activity
- Tracking the activity of competitors for signs of price and product catalog matching
- Enforcing site terms and conditions that stop malicious web scraping
- Employing bot protection capabilities with deep behavioral analysis to pinpoint bad bots and prevent web scraping



USING MACHINE LEARNING AS A SOLUTION

STEP 1: DATA PROCESSING AND FEATURE EXTRACTION

The given access log data will first be parsed and the following fields/ features will be extracted as applicable:

- Timestamp - Time at which the API access request was made
- IP Address - IP address of the client
- Country - Country of origin of the request
- HTTP request referrer- Address of the site from where the request originated.
- HTTP request method- GET, POST, etc.
- URL - Address of data requested by client.
- Number of parameters in the query.

Other features may also be extracted depending on the log format for eg. User ID, Status Code, User Agent String etc.

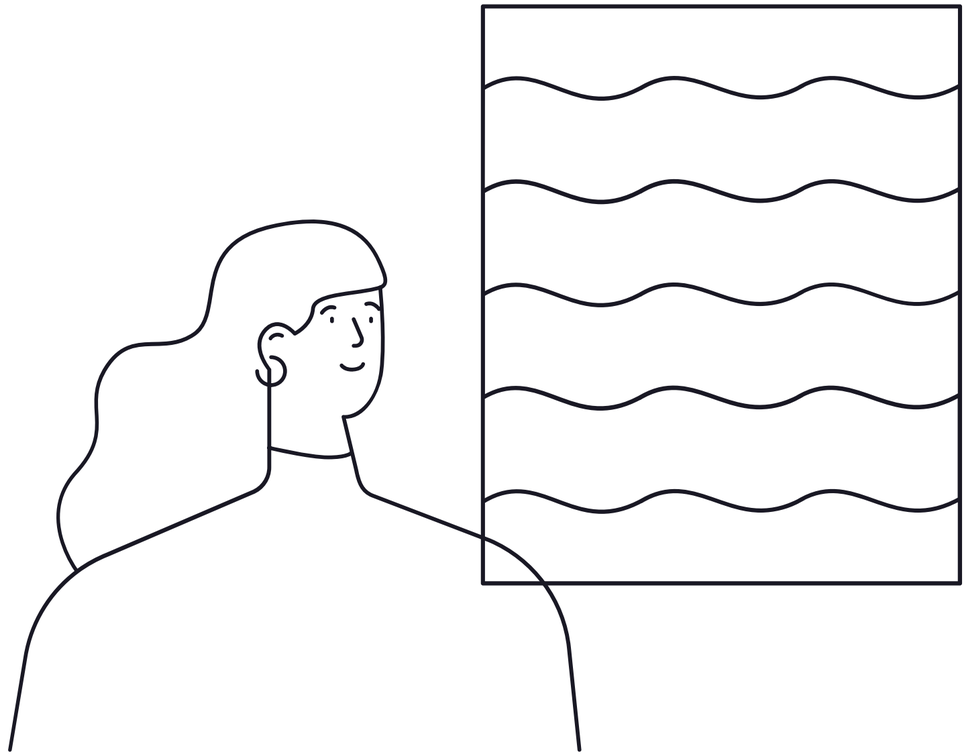
FEATURE ENGINEERING

New features will be created to allow the models to draw better inferences. The new features we plan on creating are :

1. Number of requests per user
2. Number of locations/countries associated with a particular user
3. Number of unique IP addresses associated with a user
4. Average rate of requests sent by a user
5. Number of API endpoints accessed by a user
6. Number of unique URI requests.
7. Number of failed status code requests made by the user (Eg: of the form 4xx, 5xx for HTTP requests : 400-Bad Request, 502- Bad Gateway)
8. Multiple user IDs having same User Agent String.(Fake account creation detection)

Idea behind these features:

1. A particular user account sending requests from multiple locations will indicate possible credential stuffing and account takeover.
2. Similar reasoning will apply to the number of IP addresses as well. However, even non abuse (negative) cases can use multiple IPs. Hence, a possibly large number of IPs associated with an account could indicate credential stuffing.
3. Bot content scrapers often send a large number of requests and hence the feature for average rate of requests and number of requests per user.
4. Bots scraping the internet will typically use a large number API endpoints.
5. API Misuse cases i.e developers committing mistakes while trying to access the API will typically have a large number of failed status codes.
6. Multiple user IDs being created by the same user agent detects creation of fake accounts as, usually, multiple accounts are created by the same malicious user.



DATA PREPARATION

If the dateset permits us to use supervised machine learning algorithms we will label the data as follows:
After extracting the wanted features from the raw HTTP server log files, we can label it using two labels: 1 to indicate API abuse/misuse and 0 for normal behavior or no threat.

Labeling should normally be done manually. In API abuse detection, it can be done automatically using a function that will look for specific patterns in each URL and decide about the type of attack.

The retrieved data can now be used to train our classifiers.

ML MODELS TO BE USED

Instead of relying on one model, we intend to test the data on a couple of models like:

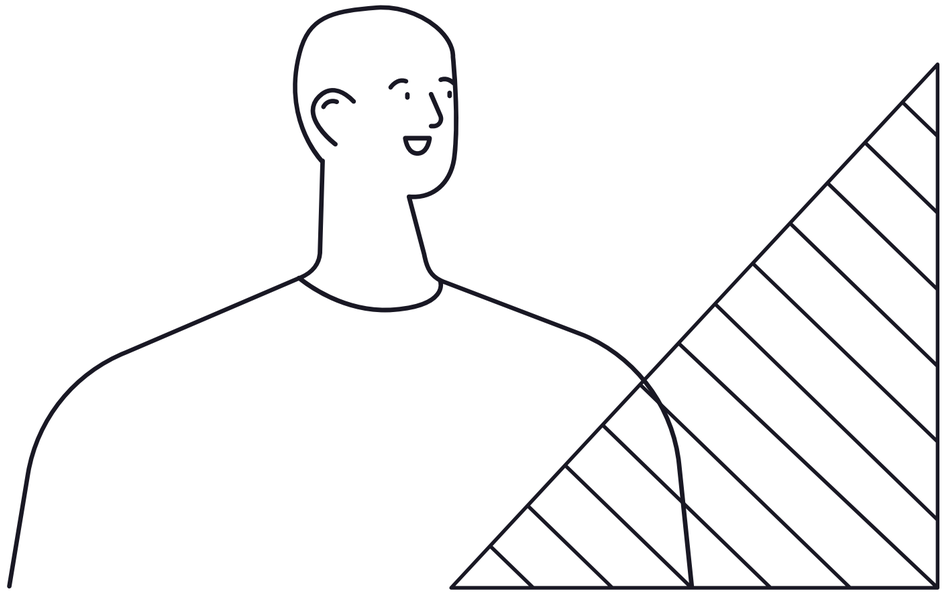
1. K-Means
2. Hierarchical Clustering

Depending on the size of access log data available, we will consider manual/programmatic labelling on data.
Labeling will be done looking at common malicious patterns in URL
If labelling is feasible, we intend to run the following supervised learning models:

1. Support Vector Machine
2. Random Forest

Many experiments show that Random Forest is a suitable classifier for web API request data as it is maintains a low false positive rate.

Programming Language of choice : Python
Libraries: Scikit Learn, Matplotlib, Numpy, Pandas, Seaborn for visualization.



AT LAST WE WILL BE COMPARING THE RESULTS OBTAINED FROM THE MENTIONED MODELS AND SOME OTHER MODELS DEPENDING ON THE PROVIDED DATASET!