How does it work?

Session Identification

Session is identified using the combination of IP and User Agent.

- IP is identified using `$_SERVER['HTTP_TRUE_CLIENT_IP']`
- User agent is identified using `$_SERVER['HTTP_USER_AGENT']`

For every request received at Shiksha.com, we'll call CABIS API:

```
/sessiontracker?reqtype=getsid&ip={{ IP }}&ua={{ UA }}&uri={{ URI }}&ref={{ REFERER
}}
```

URI: Current Page URI
REFERER: Page Referer

Using the combination of IP and User Agent, CABIS will generate a session Id, which is used to uniquely identify a user for all subsequent requests.

For all subsequent requests, CABIS keeps storing the data and after the configured no. of requests have been received, bot detection algo is triggered, after which it'll be able to tell whether it's a bad bot.

Responses

When we send the above mentioned request to CABIS, it responds with a status which tells the state of the user. The status can have the following values:

- **Negative:** Bad user/bot
- **Positive:** User user/bot
- **Zero:** Either the algo has not yet triggered (e.g. waiting for minimum no. of requests) or the algo was not able to ascertain whether it's a good or bad user

Bot Detection Algo

CABIS runs a battery of test to determine the status of current user. Once the minimum no. of requests is reached, detection algo kicks in. The tests are evaluated in the following order:

1. Bad User Agent

The very first check is whether request is made using a bad user agent. CABIS maintains a list of bad user agents and checks user's agent against this list. This list mostly comprises user agents provided by standard crawling scripts (in python, java etc.) and user is too lazy to change it before running the script.

We can also add our own bad user agent.

Response: **-3** (= UA BLACKLISTED)

2. InfoEdge IPs/Other Whitelisted IPs

Next test is whether the user IP is an InfoEdge IP or any other whitelisted IP. The list of InfoEdge/Whitelisted IPs is maintained in a config file and new IPs can be added using an API call.

Response: **2** (= IP WHITELISTED)

3. Blacklisted IPs

Next the user IP is checked against a list of blacklisted IPs. The list of blacklisted IPs is maintained in a config file and new IPs can be added using an API call.

Response: **-2** (= IP BLACKLISTED)

4. Good Bot

The check for good bot is done by matching against standard user agents of well known good bots (google, bing etc.).

As an extra safety measure, it also verified that the user's IP actually belongs the well known bot's servers using reverse DNS.

If these tests are passed, then it's a good bot.

Response: **4** (= GOOD BOT)

5. Bad Bot/User

The final check is done to identify bad browsing behavior usually exhibited by crawlers. Following checks are done:

- **Hits per minute:** If user's hits/min exceeds the configured value (e.g. more than 100 requests/min), then it might be a bad user

- **Empty referer ratio:** The ratio of in how many requests an empty referer was received out of the total requests is matched against a configured value (e.g. 50%). If the value exceeds the configured number, it might be a bad bot.

- **GET/POST ratio:** This is not currently implemented in CABIS, but can be a good extension. A valid user will have a good GET/POST ratio, whereas a bot/crawler will mostly make GET requests.

Response: **-1** (= BAD BOT)

Response Values at a Glance

Response Value	Meaning
-3	User Agent Blacklisted
-2	IP Blacklisted
-1	Bad Bot
0	Uncertain
1	User (Showing valid browsing pattern)
2	IP Whitelisted
3	User Agent Whitelisted
4	Good Bot

Javascript Integration

CABIS also has javascript integration where we place a small javascript in header/footer of our website pages. When a user opens a page, a beacon is fired to CABIS API, where it registers that this is a valid user browsing the website from a web browser (**Response = 4**).

We'll need to explore this more.

Action for Bad Bots

When we detect a bad bot, we'll redirect the user to a captcha page (implementing Google ReCaptcha). User has to successfully fill the captcha in order to continue browsing the site.

Once the user successfully fills the captcha, we'll call CABIS API to whitelist the user. From that point onwards, that user will always be a good user.

Currently there is no TTL for this setting. Ideally the status (Good or Bad) should be valid for a certain time period and then should expire. After that, the detection algo should kick-in again to refresh the status of the user. This is currently not implemented and can be a good extension.

Shiksha Configuration

Following the points to be discussed:

1. Where should the detection code be placed? One place is in hooks where it'll be triggered for all requests (main as well as ajax/beacon requests). Other is in common view header, so that it'll be triggered for only main requests and not ajax/beacon.

2. What should be the starting values for the following:

    a. Minimum no. of requests required to trigger algo
    b. Requests/minute threshold
    c. Empty referer ratio threshold

2(a) and 2(b) will depend on 1 (e.g. whether we want to include ajax/beacon calls in detection)