PHP Programming Exercise

Your goal is to write a simple web application firewall in PHP that takes rules from a YAML file to inspect the headers of an HTTP request and decides to either blacklist (403) them, ratelimit (429) them or whitelist (just let the request through) them.

You can use the latest versions of either Laravel, Symfony, Lumen or Slim to build this. Using these will ensure you code to PSR-7 and / or PSR-15 compliant requests.

Sample YAML

Here is the sample YAML you will need to use:

```yaml
# Note that values separated with a comma are always OR and each of the global
keys are always AND
blacklist:
  - name: my blacklist
    headers:
      request:
        X-FORWARDED_FOR: [123.456.78.9, 78.99.90.3]
        FORWARDED: [123.456.78.9, 78.99.90.3]
        USER-aGENT: [Mozilla/5.0, python-requests/2.8]
      server:
        rEQUEST_URI: [path/one, path/two]
        QUERY_STRING: [one=yes&two=no&three=maybe, another=0&someother=1]

whitelist:
  - name: my whitelist
    headers:
      request:
          X_FORWARDED_FOR: [123.456.78.9, 78.99.90.3]
          FORWARDED: [123.456.78.9, 78.99.90.3]
          X_FORWARDED: [123.456.78.9, 78.99.90.3]
          X_CLUSTER_CLIENT_IP: [123.456.78.9, 78.99.90.3]
          CLIENT_IP: [123.456.78.9, 78.99.90.3]
          USER_AGENT: [Mozilla/5.0, python-requests/2.8]
          REFERER: [http://something.com, 'something else']
          COOKIES: [cookie_one, another_cookie]
      server:
          REQUEST_URI: [path/one, path/two]
          QUERY_STRING: [one=yes&two=no&three=maybe]

ratelimit:
  - name: limiter
    headers:
      request:
          X_FORWARDED_FOR: [123.456.78.9, 78.99.90.3]
          FORWARDED: [123.456.78.9, 78.99.90.3]
          X_FORWARDED: [123.456.78.9, 78.99.90.3]
          X_CLUSTER_CLIENT_IP: [123.456.78.9, 78.99.90.3]
          CLIENT_IP: [123.456.78.9, 78.99.90.3]
          USER_AGENT: [Mozilla/5.0, python-requests/2.8]
          REFERER: [http://something.com, 'something else']
          COOKIES: [cookie_one, another_cookie]
```

```
      server:
          REQUEST_URI: [path/one, path/two]
          QUERY_STRING: [one=yes&two=no&three=maybe]
    limit:
      rate: 1000
      time: 3600 #60 = 1 minute, 3600 = 1 hour, 86400 = 1 day
  - name: another limiter # required
    headers:
        request:
            X_FORWARDED_FOR: [123.456.78.9, 78.99.90.3]
            FORWARDED: [123.456.78.9, 78.99.90.3]
            X_FORWARDED: [123.456.78.9, 78.99.90.3]
            X_CLUSTER_CLIENT_IP: [123.456.78.9, 78.99.90.3]
            CLIENT_IP: [123.456.78.9, 78.99.90.3]
            USER_AGENT: [Mozilla/5.0, python-requests/2.8]
            REFERER: [http://something.com, 'something else']
            COOKIES: [cookie_one, another_cookie]
        server: # required
            REQUEST_URI: [path/one, path/two]
            QUERY_STRING: [one=yes&two=no&three=maybe]
    limit:
      rate: 1000
      time: 3600 #60 = 1 minute, 3600 = 1 hour, 86400 = 1 day
```

How to interpret the YAML

The below example explains how to read the YAML for the blacklist above. The whitelist and ratelimit can be read the same way:

```
// The below is pseudo code

if the request header
    x-forwarded-for contains 123.456.78.9 OR 78.99.90.3
    AND
    forwarded contains 123.456.78.9 OR 78.99.90
    AND
    user-agent contains Mozilla/5.0 OR python-requests/2.8
AND the server header contains
    request-uri contains path/one OR path/two
    AND
    query-string contains one=yes&two=no&three=maybe OR another=0&someother=1
THEN
    this request is blacklisted
ELSE
    this request is not blacklisted
```

How to Work on the Exercise

1. Solve one problem at a time - don't try to address blacklisting, whitelisting and ratelimiting at the same time.
2. If you have enough time, build the appropriate HTTP responses to let the user know they have been blacklisted or ratelimited.
3. Write unit tests

4. Feel free to use any tools you want or need

Designing a Production Ready System for the Web Application Firewall

You are going to be deploying the above firewall in a production environment. Your firewall should be able to handle ~1000 requests a second.

Talk us through the design of this system in production. How would you handle hostnames, SSL termination, load balancing. Which webserver would you use? How would you size instances or containers?