Hacking the Elasticsearch Database

Shir Tamari • Elasticsearch Mexico Meetup • 16.02.2017

About me

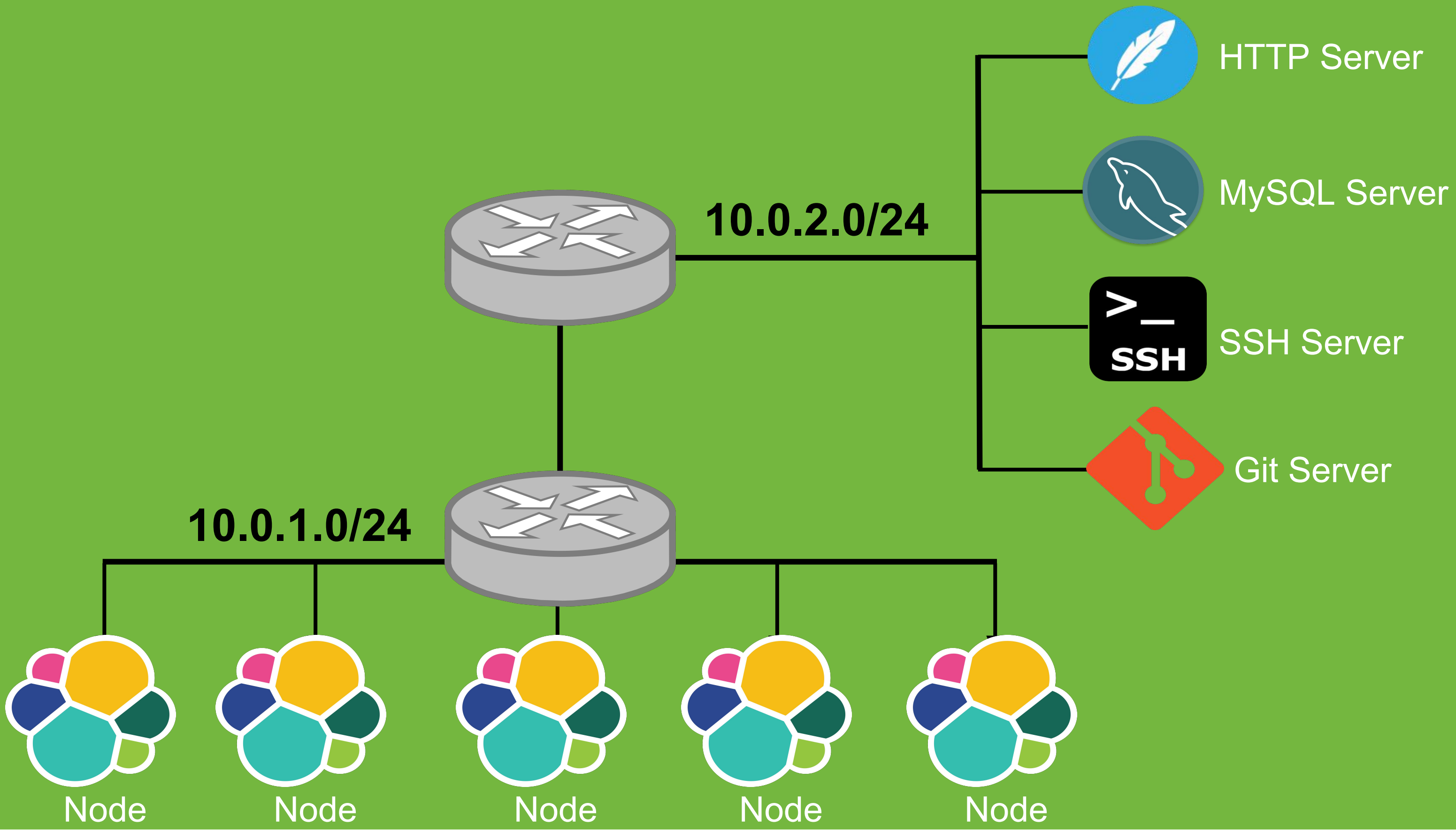
- Hacker since the age of 12
- 2009-2014 -> Army service as a software developer
- 2015-2016 -> Technology researcher at NorthBit (Acquired by Magic Leap)
- 2016-2017 -> Co Founder, VP R&D at Stablewave (Security for containers)

Overview

- Authentication issues
- Server Side Request Forgery
- Query Injection
- Mitigation Summary

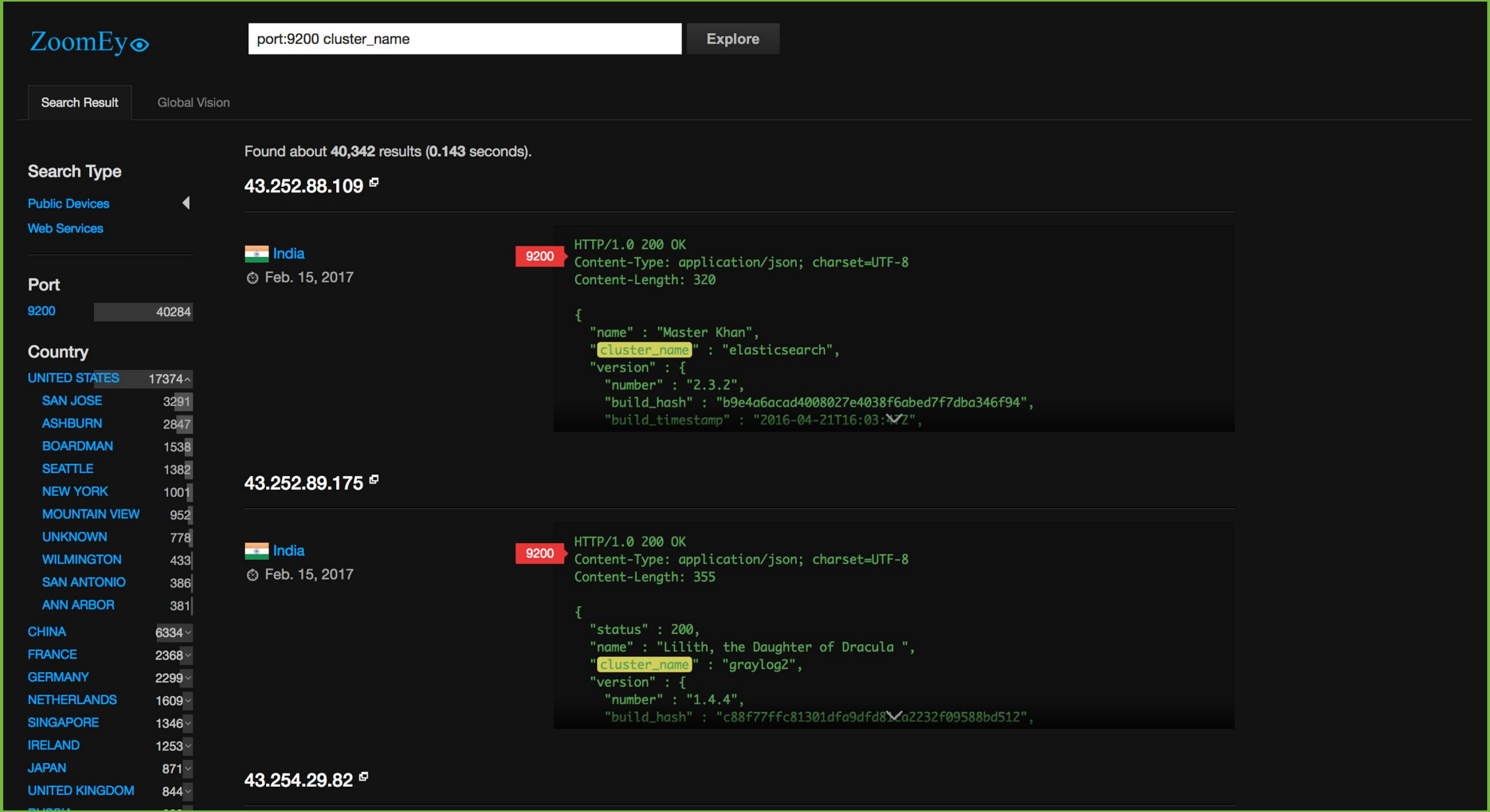
Basic Network Configuration

- Listen on **localhost** by default
- Port range is 9200-9300
- Support HTTP and Transport Protocols
- **Node's port should be exposed outside in order to be a part of a scalable cluster**


HTTP Server
MySQL Server
10.0.2.0/24
SSH
SSH Server
Git Server
10.0.1.0/24
Node
Node
Node
Node
Node

Authentication

- No built-in authentication is available
- No built-in permissions management


ZoomEye

port:9200 cluster_name

Explore

Search Result   Global Vision

Found about **40,342** results (**0.143** seconds).

Search Type

Public Devices

Web Services

Port

9200	40284

Country

UNITED STATES	17374
SAN JOSE	3291
ASHBURN	2847
BOARDMAN	1538
SEATTLE	1382
NEW YORK	1001
MOUNTAIN VIEW	952
UNKNOWN	778
WILMINGTON	433
SAN ANTONIO	386
ANN ARBOR	381
CHINA	6334
FRANCE	2368
GERMANY	2299
NETHERLANDS	1609
SINGAPORE	1346
IRELAND	1253
JAPAN	871
UNITED KINGDOM	844

43.252.88.109

India

Feb. 15, 2017

9200

```
HTTP/1.0 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 320

{
  "name" : "Master Khan",
  "cluster_name" : "elasticsearch",
  "version" : {
    "number" : "2.3.2",
    "build_hash" : "b9e4a6acad4008027e4038f6abed7f7dba346f94",
    "build_timestamp" : "2016-04-21T16:03:47Z",
```

43.252.89.175

India

Feb. 15, 2017

9200

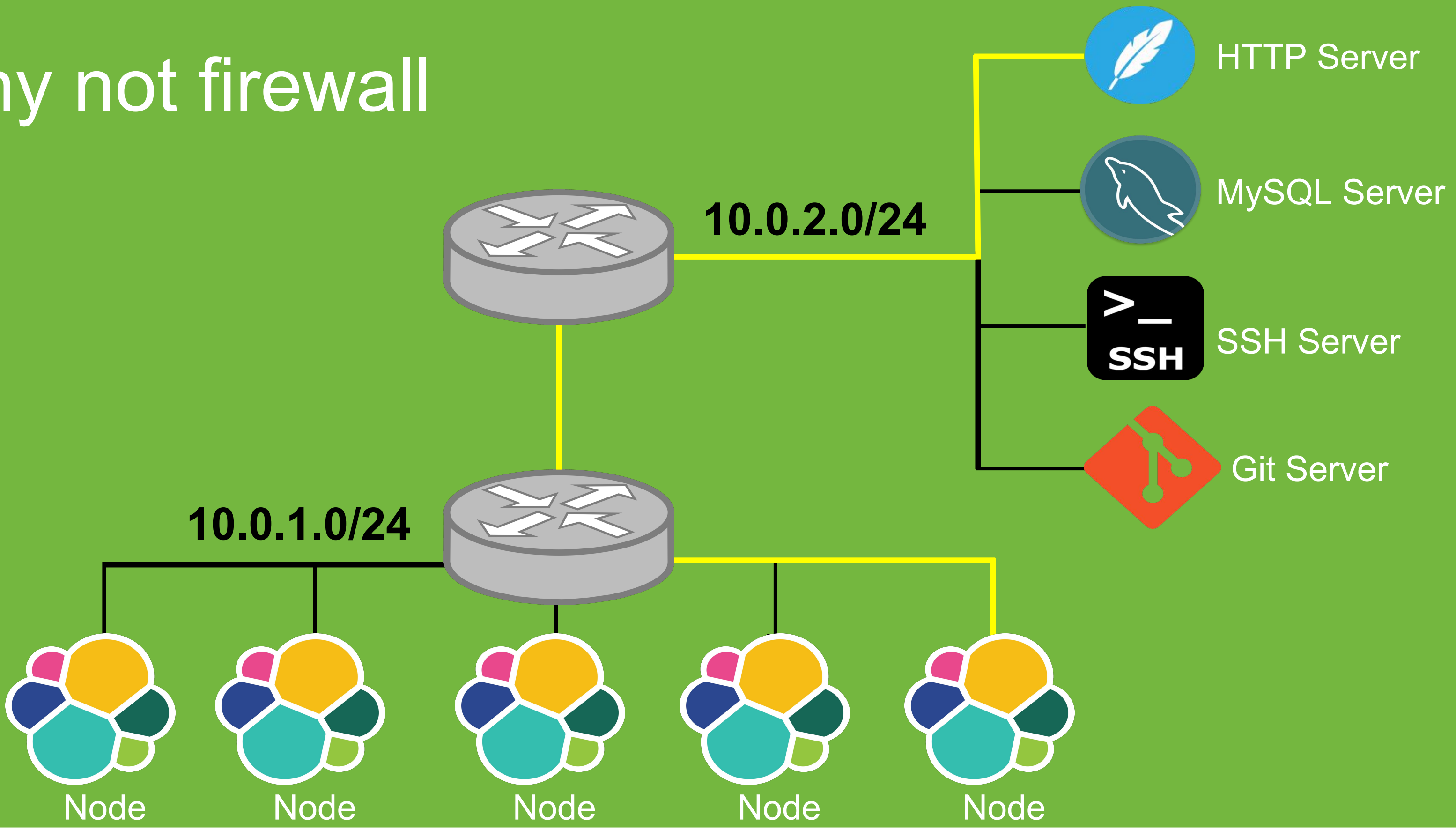
```
HTTP/1.0 200 OK
Content-Type: application/json; charset=UTF-8
Content-Length: 355

{
  "status" : 200,
  "name" : "Lilith, the Daughter of Dracula ",
  "cluster_name" : "graylog2",
  "version" : {
    "number" : "1.4.4",
    "build_hash" : "c88f77ffc81301dfa9dfd81ca2232f09588bd512",
```

43.254.29.82

Solutions

- **Firewall**
- **X-Pack** (Official, not free)**-**
  - An Elastic Stack extension that bundles security, alerting, monitoring, reporting, and graph capabilities
- **Search Guard** - (Not-Official, **Free**)
  - https://floragunn.com/searchguard/

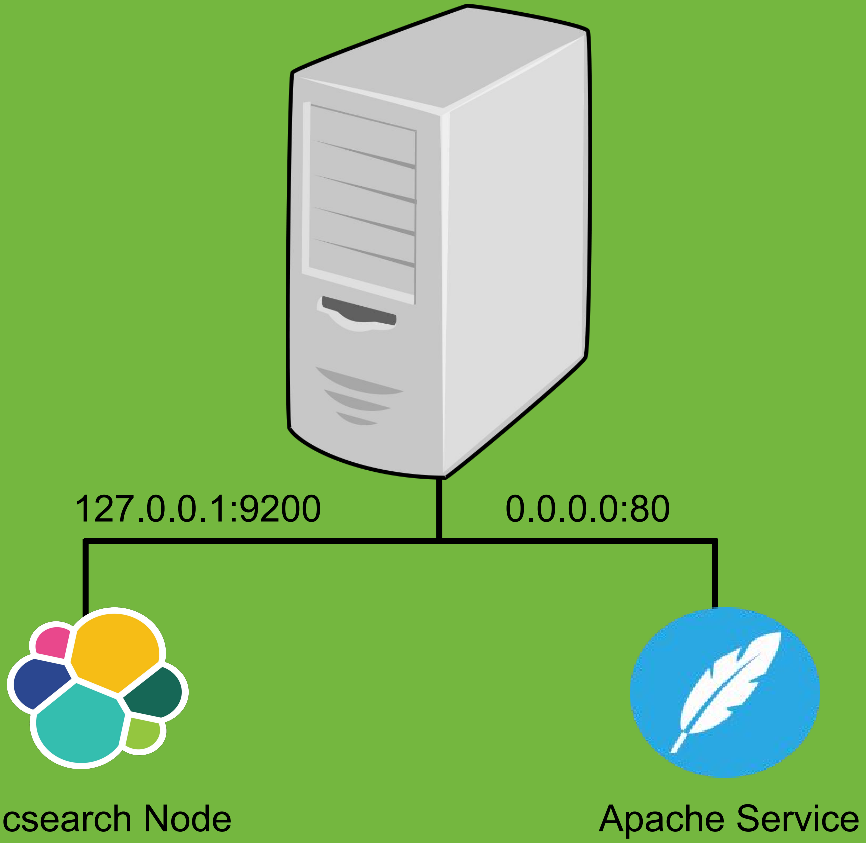
Why not firewall

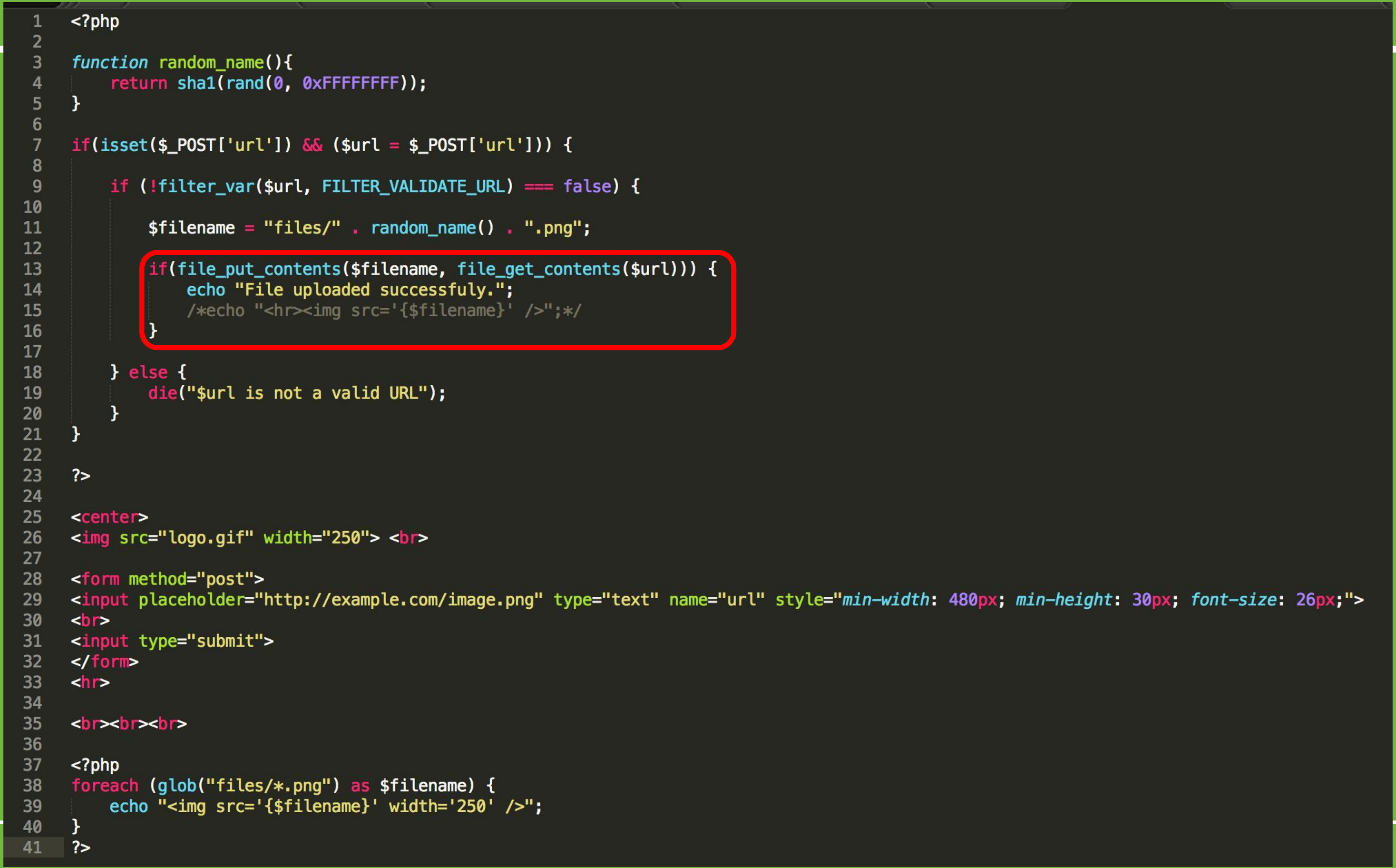


Server Side Request Forgery *(SSRF)* -

A vulnerability that appears when an attacker has the ability to create requests from the vulnerable server.

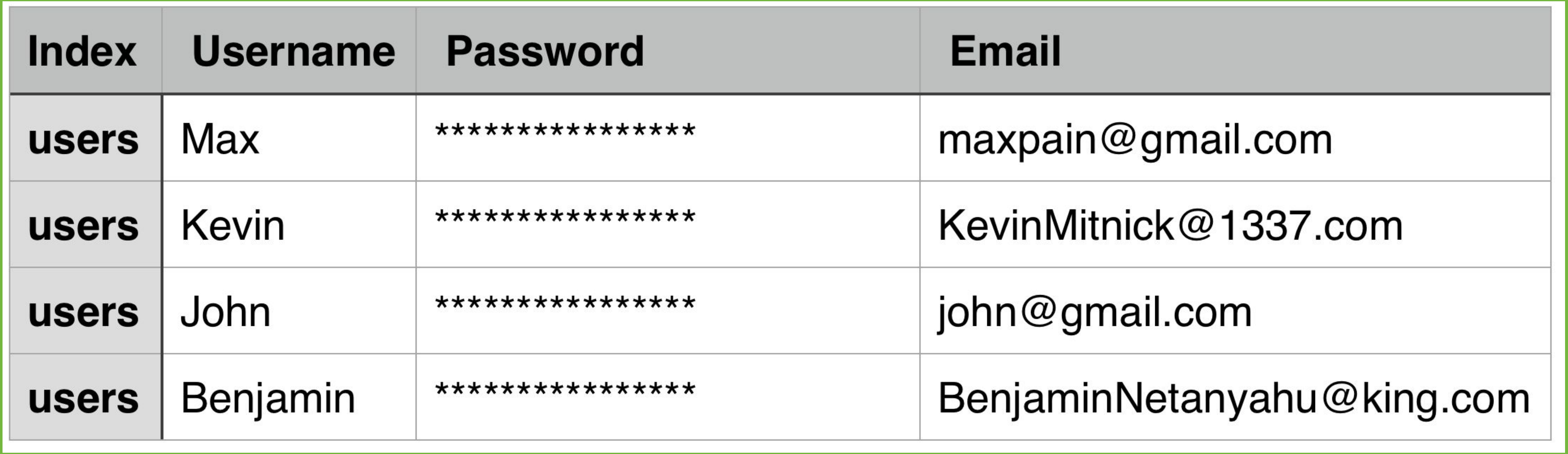
- Access limited resources
- Bypass firewalls


127.0.0.1:9200
0.0.0.0:80
Elasticsearch Node
Apache Service



```php
<?php

function random_name(){
    return sha1(rand(0, 0xFFFFFFFF));
}

if(isset($_POST['url']) && ($url = $_POST['url'])) {

    if (!filter_var($url, FILTER_VALIDATE_URL) === false) {

        $filename = "files/" . random_name() . ".png";

        if(file_put_contents($filename, file_get_contents($url))) {
            echo "File uploaded successfuly.";
            /*echo "<hr><img src='{$filename}' />";*/
        }

    } else {
        die("$url is not a valid URL");
    }
}

?>

<center>
<img src="logo.gif" width="250"> <br>

<form method="post">
<input placeholder="http://example.com/image.png" type="text" name="url" style="min-width: 480px; min-height: 30px; font-size: 26px;">
<br>
<input type="submit">
</form>
<hr>

<br><br><br>

<?php
foreach (glob("files/*.png") as $filename) {
    echo "<img src='{$filename}' width='250' />";
}
?>
```

Database



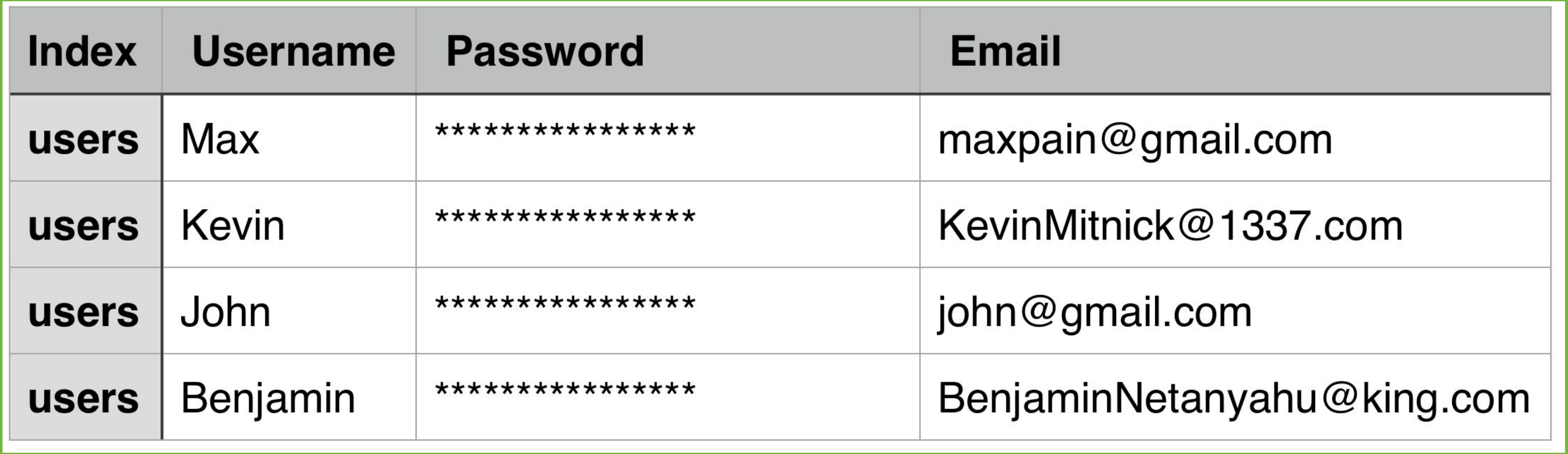
Index	Username	Password	Email
users	Max	************************	maxpain@gmail.com
users	Kevin	************************	KevinMitnick@1337.com
users	John	************************	john@gmail.com
users	Benjamin	************************	BenjaminNetanyahu@king.com

Demo

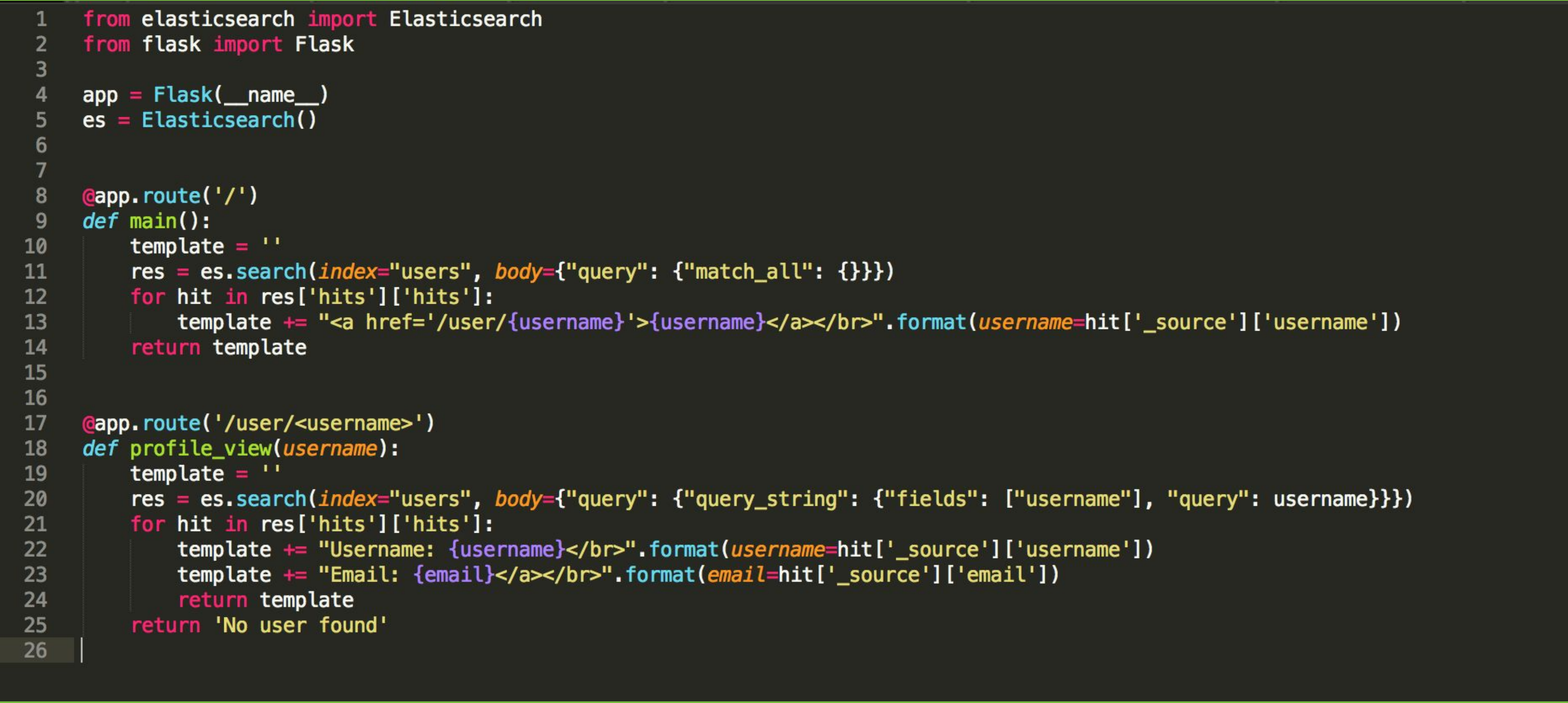
NoSQL Injection

- Syntax is really elastic
- Lot of hidden features

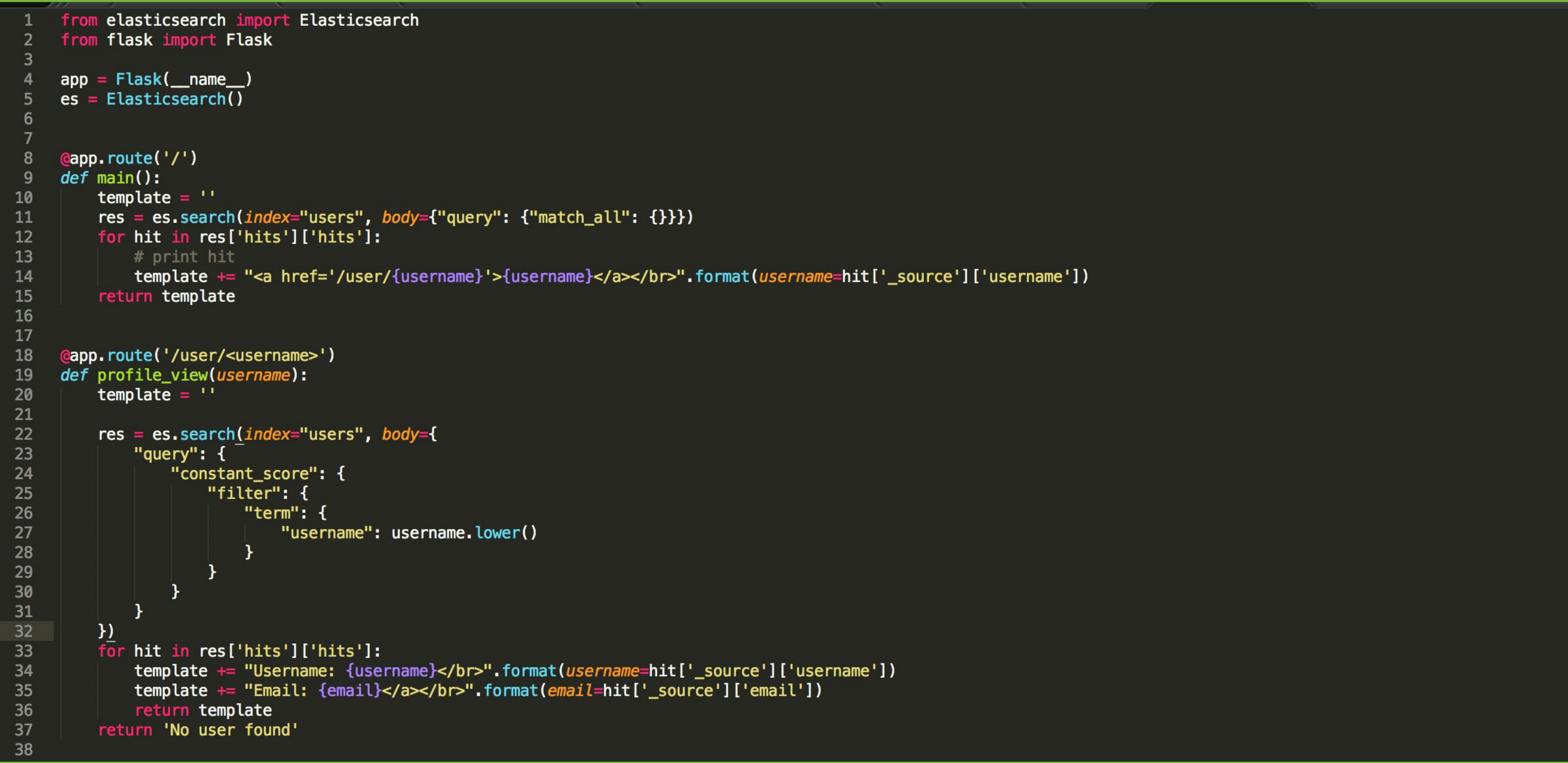
Database



Index	Username	Password	Email
users	Max	****************	maxpain@gmail.com
users	Kevin	****************	KevinMitnick@1337.com
users	John	****************	john@gmail.com
users	Benjamin	****************	BenjaminNetanyahu@king.com



```
from elasticsearch import Elasticsearch
from flask import Flask

app = Flask(__name__)
es = Elasticsearch()


@app.route('/')
def main():
    template = ''
    res = es.search(index="users", body={"query": {"match_all": {}}})
    for hit in res['hits']['hits']:
        template += "<a href='/user/{username}'>{username}</a></br>".format(username=hit['_source']['username'])
    return template


@app.route('/user/<username>')
def profile_view(username):
    template = ''
    res = es.search(index="users", body={"query": {"query_string": {"fields": ["username"], "query": username}}})
    for hit in res['hits']['hits']:
        template += "Username: {username}</br>".format(username=hit['_source']['username'])
        template += "Email: {email}</a></br>".format(email=hit['_source']['email'])
        return template
    return 'No user found'

```



```python
from elasticsearch import Elasticsearch
from flask import Flask

app = Flask(__name__)
es = Elasticsearch()


@app.route('/')
def main():
    template = ''
    res = es.search(index="users", body={"query": {"match_all": {}}})
    for hit in res['hits']['hits']:
        # print hit
        template += "<a href='/user/{username}'>{username}</a></br>".format(username=hit['_source']['username'])
    return template


@app.route('/user/<username>')
def profile_view(username):
    template = ''

    res = es.search(index="users", body={
        "query": {
            "constant_score": {
                "filter": {
                    "term": {
                        "username": username.lower()
                    }
                }
            }
        }
    })
    for hit in res['hits']['hits']:
        template += "Username: {username}</br>".format(username=hit['_source']['username'])
        template += "Email: {email}</a></br>".format(email=hit['_source']['email'])
        return template
    return 'No user found'

```

Demo

Conclusion

1. Use Firewall!
2. You better protect your elastic node with a password
3. Read the Elasticsearch documentation carefully when executing user's input.

MongoDB

- No authentication by default
    - Authentication module built-in :)

- SSRF vulnerability could affect the db in case of net.http.enabled = True

MongoDB

```
If(db->users->find({username: $_POST[‘user’], password: $_POST[‘password’]})){
        //user is logged in
}

POST /login
user[$gt]=&password[$gt]=



db.users.find({username: {$gt: “”}, password: {$gt: “”}})
```