Infrastucture of The WEB

It's !tutorial;

Muhammad Falak R Wani

Agenda

- HTTP
- GET
- POST
- Web Servers
- Static Sites
- Dynamic Sites
- Role of Linux/Unix
- Web Frameworks
- Some Real World Case Studies
- Choices For Deployment (python, node.js, golang)

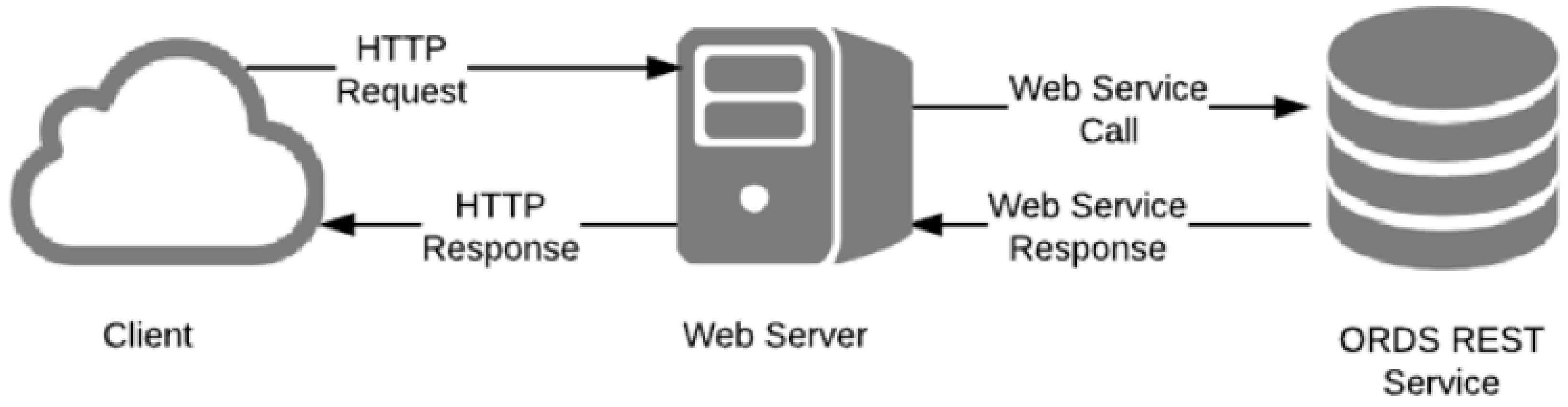
```
This course has no language constraint.
```

HTTP

A simple Text Based protocol

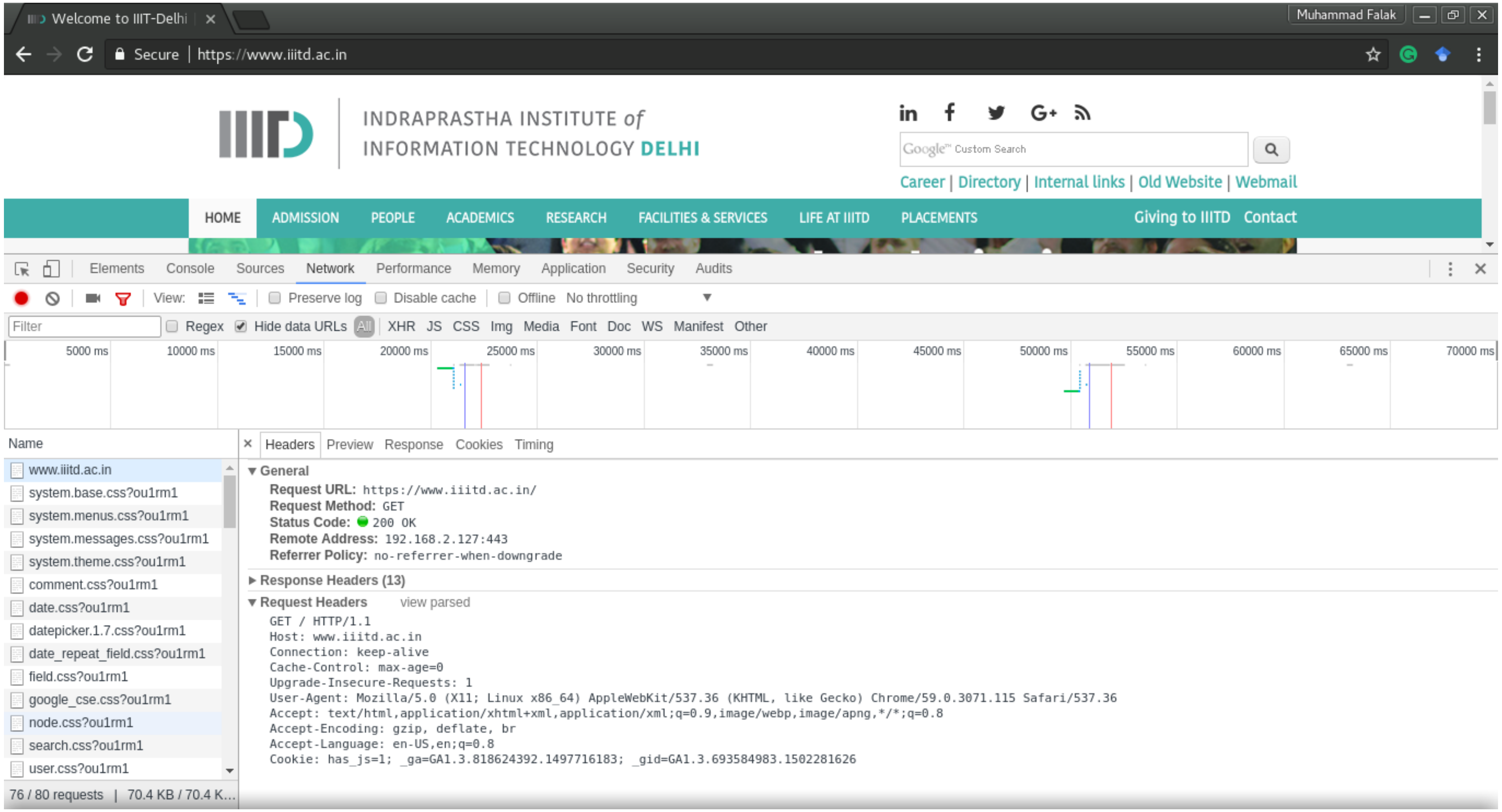
- Server
- Client

Client: Initiates a *request*
Server: Responds with a *response*



GET

```
www.iiitd.ac.in
```


Welcome to IIIT-Delhi
Muhammad Falak
Secure | https://www.iiitd.ac.in
INDRAPRASTHA INSTITUTE of INFORMATION TECHNOLOGY DELHI
Google™ Custom Search
Career | Directory | Internal links | Old Website | Webmail
HOME ADMISSION PEOPLE ACADEMICS RESEARCH FACILITIES & SERVICES LIFE AT IIITD PLACEMENTS Giving to IIITD Contact
Elements Console Sources Network Performance Memory Application Security Audits
View: Preserve log Disable cache Offline No throttling
Filter Regex Hide data URLs All XHR JS CSS Img Media Font Doc WS Manifest Other
5000 ms 10000 ms 15000 ms 20000 ms 25000 ms 30000 ms 35000 ms 40000 ms 45000 ms 50000 ms 55000 ms 60000 ms 65000 ms 70000 ms
Name
Headers Preview Response Cookies Timing
76 / 80 requests | 70.4 KB / 70.4 K...

Name:
- www.iiitd.ac.in
- system.base.css?ou1rm1
- system.menus.css?ou1rm1
- system.messages.css?ou1rm1
- system.theme.css?ou1rm1
- comment.css?ou1rm1
- date.css?ou1rm1
- datepicker.1.7.css?ou1rm1
- date_repeat_field.css?ou1rm1
- field.css?ou1rm1
- google_cse.css?ou1rm1
- node.css?ou1rm1
- search.css?ou1rm1
- user.css?ou1rm1

▼ General

Request URL: `https://www.iiitd.ac.in/`
Request Method: `GET`
Status Code: ● `200 OK`
Remote Address: `192.168.2.127:443`
Referrer Policy: `no-referrer-when-downgrade`

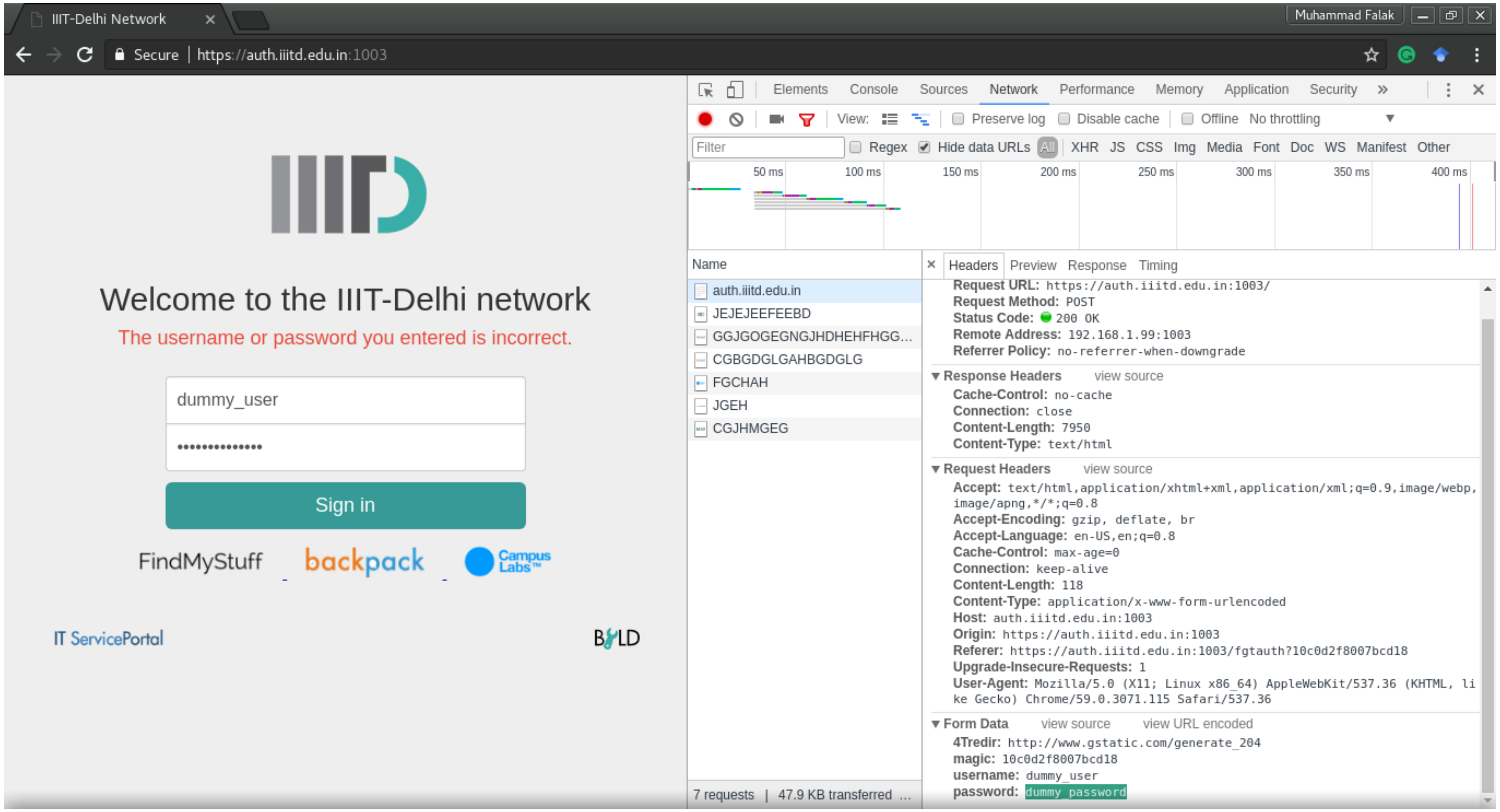
▶ Response Headers (13)

▼ Request Headers   view parsed

```
GET / HTTP/1.1
Host: www.iiitd.ac.in
Connection: keep-alive
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/59.0.3071.115 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.8
Cookie: has_js=1; _ga=GA1.3.818624392.1497716183; _gid=GA1.3.693584983.1502281626
```

POST

When we login in iiitd


IIIT-Delhi Network
Secure | https://auth.iiitd.edu.in:1003

Welcome to the IIIT-Delhi network
The username or password you entered is incorrect.
dummy_user
Sign in

```
auth.iiitd.edu.in
JEJEJEEFEEBD
GGJGOGEGNGJHDHEHFHGG...
CGBGDGLGAHBGDGLG
FGCHAH
JGEH
CGJHMGEG
7 requests | 47.9 KB transferred ...
```

```
Request URL: https://auth.iiitd.edu.in:1003/
Request Method: POST
Status Code: 200 OK
Remote Address: 192.168.1.99:1003
Referrer Policy: no-referrer-when-downgrade
▼ Response Headers      view source
   Cache-Control: no-cache
   Connection: close
   Content-Length: 7950
   Content-Type: text/html
▼ Request Headers      view source
   Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,
   image/apng,*/*;q=0.8
   Accept-Encoding: gzip, deflate, br
   Accept-Language: en-US,en;q=0.8
   Cache-Control: max-age=0
   Connection: keep-alive
   Content-Length: 118
   Content-Type: application/x-www-form-urlencoded
   Host: auth.iiitd.edu.in:1003
   Origin: https://auth.iiitd.edu.in:1003
   Referer: https://auth.iiitd.edu.in:1003/fgtauth?10c0d2f8007bcd18
   Upgrade-Insecure-Requests: 1
   User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, li
   ke Gecko) Chrome/59.0.3071.115 Safari/537.36
▼ Form Data      view source      view URL encoded
   4Tredir: http://www.gstatic.com/generate_204
   magic: 10c0d2f8007bcd18
   username: dummy_user
   password: dummy_password
```

HTML

It's not a language, but markup.

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="UTF-8" />
        <meta name="viewport" content="width=device-width" />
        <title>Hello</title>
    </head>
    <body>
        Hello World
    </body>
</html>
```

Its a static blob of text which your browser can fetch and render

HTML Forms

The most basic way of getting information.

- POST METHOD
- GET METHOD

```
<form action="/my-handling-form-page" method="get">
    <div>
        <label for="name">Name:</label>
        <input type="text" id="name" name="user_name">
    </div>
    <div>
        <label for="mail">E-mail:</label>
        <input type="email" id="mail" name="user_mail">
    </div>
    <div>
        <label for="msg">Message:</label>
        <textarea id="msg" name="user_message"></textarea>
    </div>
</form>
```

GET Example

Lets send a string to a webserver via telnet

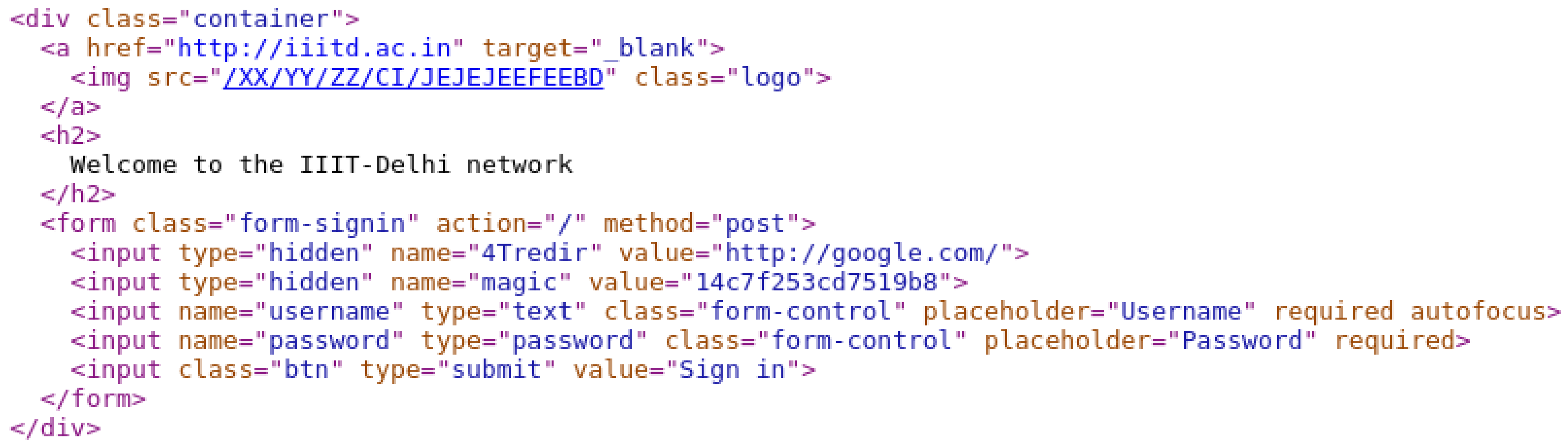
```
mfrw ➜  ~ telnet www.google.com 80
Trying 172.217.26.164...
Connected to www.google.com.
Escape character is '^]'.
GET / http/1.0
Host: www.google.com

HTTP/1.0 400 Bad Request
Content-Type: text/html; charset=UTF-8
Referrer-Policy: no-referrer
Content-Length: 1555
Date: Fri, 11 Aug 2017 06:29:12 GMT

<!DOCTYPE html>
<html lang=en>
  <meta charset=utf-8>
    <meta name=viewport content="initial-scale=1, minimum-scale=1, width=device-width">
      <title>Error 400 (Bad Request)!!1</title>
        <style>
.........
```

IIITD Login

What if we were able to use what we learnt into something productive



```html
<div class="container">
  <a href="http://iiitd.ac.in" target="_blank">
    <img src="/XX/YY/ZZ/CI/JEJEJEEFEEBD" class="logo">
  </a>
  <h2>
    Welcome to the IIIT-Delhi network
  </h2>
  <form class="form-signin" action="/" method="post">
    <input type="hidden" name="4Tredir" value="http://google.com/">
    <input type="hidden" name="magic" value="14c7f253cd7519b8">
    <input name="username" type="text" class="form-control" placeholder="Username" required autofocus>
    <input name="password" type="password" class="form-control" placeholder="Password" required>
    <input class="btn" type="submit" value="Sign in">
  </form>
</div>
```

- Send a **GET** Request to *google.com*
- If the response contains any trace of **IIIT-Delhi**
- Extract the **magic** number
- Reply with the magic number + username + password
- Print **Logout** url

IIITD Login code (golang)

```go
package main

import (
    "fmt"
    "io/ioutil"
    "net/http"
    "net/url"
)

func main() {
    res, _ := http.Get("http://www.google.com/")
    if res.Request.URL.Hostname() == "auth.iiitd.edu.in" {
        magic := res.Request.URL.RawQuery
        u := res.Request.URL.String()
        res, _ = http.PostForm(u, url.Values{
            "magic":    {magic},
            "username": {"falak16018"},
            "password": {"*********"},
        })
        defer res.Body.Close()
        body, _ := ioutil.ReadAll(res.Body)
        fmt.Println("Logout:", string(body[4816:4870]))
    }
}
```

Run

IIITD Login code (python2)

```
import requests
import urllib
import getpass

def login(username, password):
    r = requests.get('http://www.google.com') # GET

    if r.url.find('google.com') == -1 :
        magic = urllib.splitquery(r.url)[1]
         values = {
                 'username':username,
                 'password':password,
                 'magic':magic,
                 }
         r2 = requests.post(r.url, data=values) # POST
         print 'Logout: ', r2.content[4816:4870]
     else:
         print 'Already connected'

 if __name__ == '__main__':
     password = getpass.getpass("Enter Password:")
     login('falak16018', password)
```

Web Servers

Listen on a port and serves webpages

```
Default is 80/443 http/https
```

- Apache
- Lighttpd
- nginx
- Python
- node.js
- golang ... etc etc

```
By default they serve index.htm[l]
```

Static Sites

Demo time

- Apache
- Lighttpd
- nginx

Docker Demo

nginx

```
docker run -i -t -v $PWD:/usr/share/nginx/html -p 80:80 nginx
```

- -i -t : Bind an interactive terminal
- -v : Share a present directory in the container
- -p : expose a port from the container to the host machine
- nginx : The name of the image to run

```
You could also install this natively
```

Dynamic Websites

- Python
- node.js
- golang

Using other Langugaes

For python2 to start a web server

```
python2 -m SimpleHTTPServer 8080
```

For python3 to start a web server

```
python3 -m http.server  8080
```

Simple Webserver in golang

Languages such as Python, ruby, golang etc have web-servers inbuilt

By far the most high performant is for **golang**

```go
package main

import (
    "log"
    "net/http"
)

func main() {
    log.Println("[+] Started a webserver listening on port 8080")
    http.ListenAndServe(":8080", http.FileServer(http.Dir(".")))
}
```

Run

Simple Webserver in node.js

```
var http = require('http');
var fs = require('fs');
var index = fs.readFileSync('index.html');

var server = http.createServer(function (req, res) {
      res.writeHead(200, {'Content-Type': 'text/plain'});
      res.end(index);
});

server.listen(8080);

console.log("[+] Server listening on port 8080");
```

Hello World WebApp in golang

- Create a listener
- Register url handlers

```go
package main

import (
    "fmt"
    "log"
    "net/http"
)

func handler(w http.ResponseWriter, r *http.Request) {
     fmt.Fprintf(w, "Hi there, I love %s!", r.URL.Path[1:])
}

func main() {
     http.HandleFunc("/", handler)
     log.Println("[+] Server listening on 8080")
     http.ListenAndServe(":8080", nil)
}
```

Project

- A secure Banking application
- Details will be shared comming monday
- Focus on security rather than web
- We are planning to have Milestones

VM's

- Open for discussion
- Plan is to provide a bare-bones VM
- Get the infrastucture working
- Build the app
- Your peers test your implementation
- Defensive/Offensive Security

For Next Time

- Real Web Devlopment
- Using Python, node.js, golang
- Using web frameworks like Django, express ...
- Introduction to AWS, Google AppEngine, Heroku
- Details about the VM setup.
- Databases
- Caching
- Load Balancing

Thank you

Muhammad Falak R Wani
https://github.com/mfrw/talks/tree/master/webdev (https://github.com/mfrw/talks/tree/master/webdev)
falak16018@iiitd.ac.in (mailto:falak16018@iiitd.ac.in)