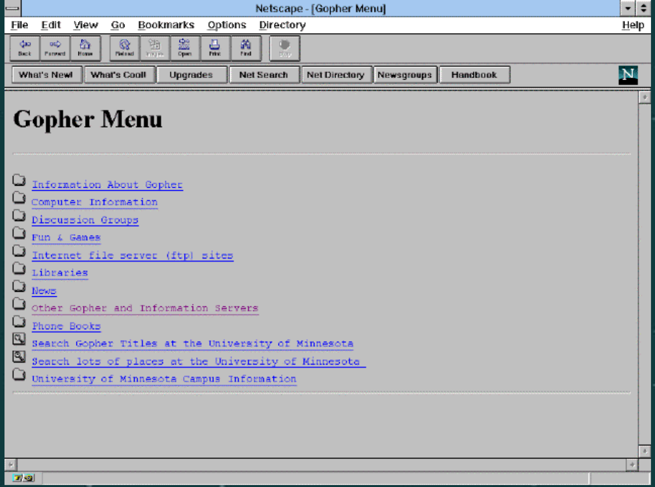
Web Application Security

Information Security – Lecture 12

Aadil Zia Khan

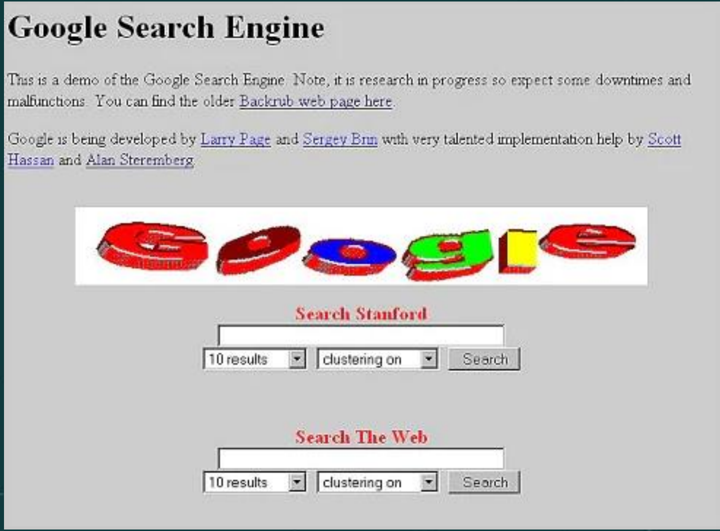
Down the Memory Lane

- The first web page went live on August 6, 1991
  - http://info.cern.ch/hypertext/WWW/TheProject.html
- Webpages were static and communication was pull based – consisted of fixed text and hyperlinks (and later images) – before dynamic pages, generated at runtime, came to the scene
- Minimum features meant securing them was easy



Gopher Menu

- Information About Gopher
- Computer Information
- Discussion Groups
- Fun & Games
- Internet file server (ftp) sites
- Libraries
- News
- Other Gopher and Information Servers
- Phone Books
- Search Gopher Titles at the University of Minnesota
- Search lots of places at the University of Minnesota
- University of Minnesota Campus Information



Google Search Engine

This is a demo of the Google Search Engine. Note, it is research in progress so expect some downtimes and malfunctions. You can find the older Backrub web page here.

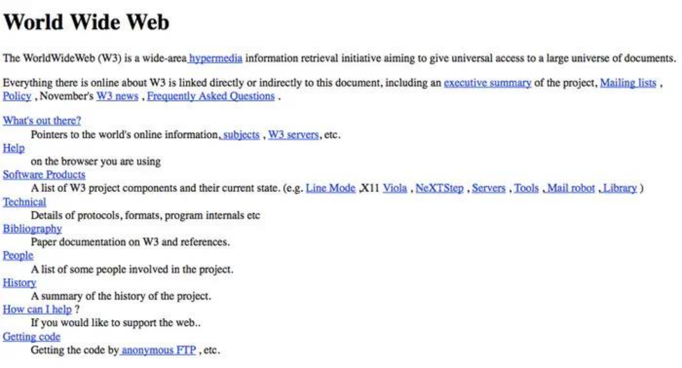
Google is being developed by Larry Page and Sergey Brin with very talented implementation help by Scott Hassan and Alan Steremberg.

Search Stanford

10 results ▾ clustering on ▾ Search

Search The Web

10 results ▾ clustering on ▾ Search



World Wide Web

The WorldWideWeb (W3) is a wide-area hypermedia information retrieval initiative aiming to give universal access to a large universe of documents.

Everything there is online about W3 is linked directly or indirectly to this document, including an executive summary of the project, Mailing lists , Policy , November's W3 news , Frequently Asked Questions .

What's out there?
Pointers to the world's online information, subjects , W3 servers, etc.

Help
on the browser you are using

Software Products
A list of W3 project components and their current state. (e.g. Line Mode ,X11 Viola , NeXTStep , Servers , Tools , Mail robot , Library)

Technical
Details of protocols, formats, program internals etc

Bibliography
Paper documentation on W3 and references.

People
A list of some people involved in the project.

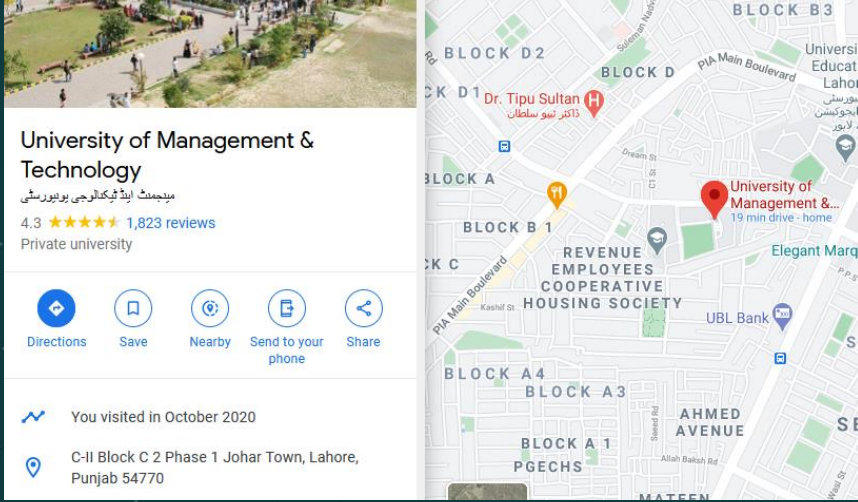
History
A summary of the history of the project.

How can I help ?
If you would like to support the web..

Getting code
Getting the code by anonymous FTP , etc.

A New Era

- Today's web technologies enable
  - Push notifications
  - Geolocation
  - Local Storage
  - And much more - unimaginable a few years ago
- Problem???
  - We are much more vulnerable
  - Developer must cater to all potential weaknesses when developing for the web


University of Management & Technology

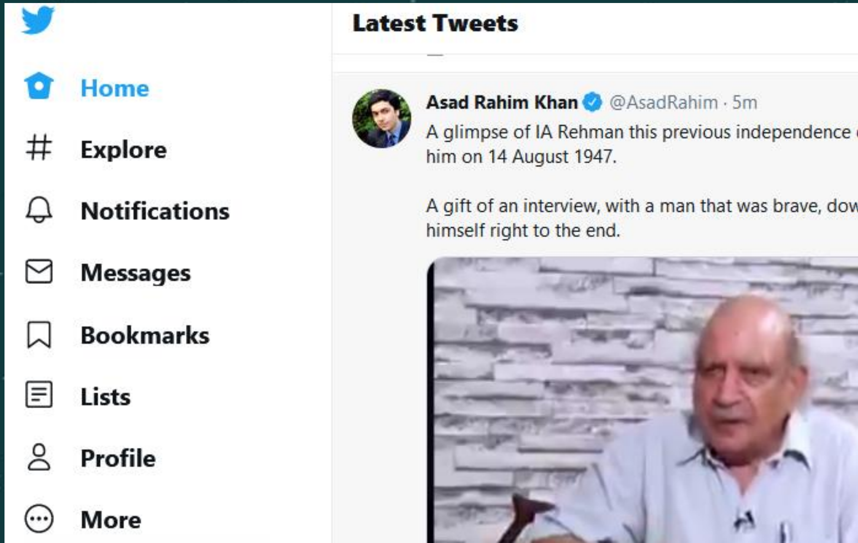
مینجمنٹ اینڈ ٹیکنالوجی یونیورسٹی

4.3 ★★★★★ 1,823 reviews

Private university

You visited in October 2020

C-II Block C 2 Phase 1 Johar Town, Lahore, Punjab 54770


Latest Tweets

Asad Rahim Khan @AsadRahim · 5m

A glimpse of IA Rehman this previous independence him on 14 August 1947.

A gift of an interview, with a man that was brave, dow himself right to the end.

Browsers, Vulnerabilities and Bug Bounty

- Over the years, many vulnerabilities have been identified in HTTP
- Companies providing web services and apps, as well as browser vendors offer a Bug Bounty program
  - Pay the hackers who are able to identify weaknesses – payment depends on the threat level
- As soon as a threat is found – the company patches it
- Some threats might have been ignored
- Websites like caniuse.com can be used to see a list of browsers and whether the threats have been patched

cURL

- cURL is a computer software project providing a library and command-line tool for transferring data using various network protocols – including HTTP
- The name stands for "Client URL"
- First released in 1997
- Try it out yourselves
  - https://reqbin.com/curl

HTTP Message Headers - Request/Response

```
GET / HTTP/1.1
Host: lms.umt.edu.pk
```

HTTP requests and responses can contain multiple headers which are used to exchange metadata and additional required information between client and server

These headers can be standard or user defined

```
HTTP/1.1 200 OK
Date: Mon, 12 Apr 2021 16:15:15 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Set-Cookie: __cfduid=d80088e74e8adbaae9d77725b0017e63d1618244114; expires=Wed,
12-May-21 16:15:14 GMT; path=/; domain=.umt.edu.pk; HttpOnly; SameSite=Lax
X-Powered-By: PHP/7.4.15
Set-Cookie: MoodleSession=6o8nap72i0f0jdigr09ieipl1l; path=/; secure
Content-Language: en
Content-Script-Type: text/javascript
Content-Style-Type: text/css
X-UA-Compatible: IE=edge
Cache-Control: private, pre-check=0, post-check=0, max-age=0, no-transform
...
```

HTTP vs HTTP2 vs HTTPS

- Started with HTTP/1.0 and HTTP/1.1
- Next came improvements
  - HTTP2 – focused towards speeding the process up
    - Uses prioritization, preemptive server push, multiplexing, binary formatting, and header compression
  - HTTPS – added security to HTTP through encryption and digital signatures
    - First ran over SSL (Secure Socket Layer), but now relies on underlying Transport Layer Security (TLS)

Transport Layer Security

- HTTPS relies on underlying TLS for confidentiality, authenticity, and integrity
- TLS connections between a client and a server has the following properties:
  - The connection is private because a symmetric-key algorithm is used to encrypt the data transmitted - keys for this symmetric encryption are generated uniquely for each connection and shared during the handshake
  - The identity of the communicating parties can be authenticated using public-key cryptography - required for the server and optional for the client
  - The connection is reliable because each message transmitted includes a message integrity check using a signed message authentication code
- TLS uses Diffie-Hellman based algorithms for key exchange, and AES for block cipher and ChaCha20-Poly1305 for stream cipher – other algorithms also available

Why Secure HTTP

- HTTP forms the backbone of modern day communication
  - Ecommerce
  - Messaging
  - Private data

HTTP GET vs POST

- GET requests - parameters are included in the URL (e.g., https://www.google.com/search?q=infosec)
- POST requests - parameters are included in the HTTP body
- GET vs POST which is better?
  - Webservers and browsers log the requests - if sensitive data is included in the URL, it could be leaked by the web server or browser
  - Webservers and browsers do not log HTTP headers or bodies, as the data to be saved would be too large - sending information through the request body, rather than through the URL, is thus safer

HTTP Security Headers

- Many HTTP security headers have been proposed to prevent known attacks and address known vulnerabilities

HTTP Strict Transport Security (HSTS)

- Run “curl www.gmail.com”
  - You’ll see “Strict-Transport-Security: max-age=31536000; includeSubDomains”

- Now run “curl lms.umt.edu.pk”
  - You’ll not see any “Strict-Transport-Security” header field

HTTP Strict Transport Security (HSTS)

- A server implements an HSTS policy by sending a “strict-transport-security: max-age=3600” header over an HTTPS connection (a different age may be specified)
  - Note a Strict-Transport-Security header sent over an insecure HTTP connection will be ignored because the connection is insecure to begin with

- When a web server issues HSTS Policy to user agents, conformant user agents behave as follows
  - Automatically turn any insecure links referencing the web application into secure links (e.g. http://example.com/some/page/ will be modified to https://example.com/some/page/ before accessing the server)
  - If the security of the connection cannot be ensured, the user agent must terminate the connection and should not allow the user to access the web application

Why Need HSTS

- A website may use HTTPS and still create insecure HTTP requests – e.g., login page is sent over HTTPS but some images may be sent over HTTP
    - End users would be exposed to Man In The Middle attack – unknowingly

- If an attacker performs a Man-In-The-Middle attack and sniffs any unencrypted traffic that flows through, then they can access those HTTP requests which may include sensitive data

- By specifying HSTS, the server ensures that all requests coming from the browser are over HTTPS

First HSTS Packet

- What happens the first time a user visits your website - there is no HSTS policy defined yet
  - Will the browser send request over HTTP or HTTPS
- Suppose an HTTPS request arrives – server is certain about the authenticity/integrity
- Suppose an HTTP request arrives – how can the server determine if the request came from the original user or from some man in the middle who replaced the actual packet (HTTPS) with his own packet (HTTP)
- To address this, browsers maintain a database of websites that enforce HSTS – e.g., hstspreload.org
  - Browsers can determine in advance that a site uses HSTS and the first interaction between clients and server will be over HTTPS – and the server would know this, thus identifying a Man in The Middle if the request is sent using HTTP

Is That It?

- There are many more security headers which cater to different attacks
  - More on it later