// WEB ENGINEERING FUNDAMENTALS — PDF 05

HTTP vs HTTPS.

How browsers and servers communicate — and how that communication becomes secure.

// TABLE OF CONTENTS

Inside this guide

01 — INTRODUCTION

Every browser-server conversation follows a set of rules

Whenever you visit a website, your browser needs a way to communicate with the server. That communication follows a set of rules known as HTTP.

Without HTTP, your browser wouldn't know how to request a webpage or how the server should respond. As the internet evolved, protecting user data became essential — passwords, payment information, emails, and personal details could not be sent in plain text. This led to HTTPS, a secure version of HTTP that encrypts communication.

```
// WHY THIS MATTERS
```

Every website, web application, API, and browser depends on these two protocols. There's no working around them.

02 — WHAT IS HTTP?

HyperText Transfer Protocol — the common language of the web

HTTP is a communication protocol that allows browsers and servers to exchange information. Your browser uses this language to ask for information, and the server uses the same language to respond. Without HTTP, websites could not be loaded.

// WHAT CAN HTTP TRANSFER?

- HTML pages
- CSS files
- JavaScript files
- Images
- Videos
- PDFs
- JSON data
- API responses

03 — WHAT IS HTTPS?

The same job as HTTP, plus a lock on the envelope

HTTPS performs the same job as HTTP but adds a security layer that encrypts data before it's transmitted. Because the data is encrypted, attackers cannot easily read or modify it while it travels across the internet. Today, almost every professional website uses HTTPS.

04 — WHY ARE HTTP AND HTTPS IMPORTANT?

A postcard vs. a locked envelope

Imagine sending a postcard — anyone who handles it can read the contents. That's similar to HTTP. Now imagine placing the same message inside a locked envelope, where only sender and receiver can read it. That's HTTPS.

// HTTPS PROTECTS

- Passwords
- Personal messages
- Login sessions
- Banking information
- Payment details
- Sensitive business data

05 — HOW HTTP WORKS

Request in, response out — five steps

1. The browser sends a request.
2. The server receives the request.
3. The server processes the request.
4. The server sends a response.
5. The browser displays the result.

This simple process powers almost every website on the internet.

06 — HOW HTTPS WORKS

Same process — with encryption established first

1. Browser requests a secure connection.
2. Server presents a digital certificate.
3. Browser verifies the certificate.
4. A secure encrypted connection is established.
5. HTTP communication begins inside the encrypted connection.

This ensures transmitted data remains private and secure.

07 — WHAT IS SSL/TLS?

The technology that makes HTTPS possible

SSL (Secure Sockets Layer) and TLS (Transport Layer Security) are technologies that make HTTPS possible. Today, TLS is the modern standard, although many people still use the term SSL out of habit.

Their responsibilities include:

- Encrypting communication
- Verifying website identity
- Protecting transmitted data

// THE PADLOCK ICON

When you see the padlock icon in your browser's address bar, TLS is what's actively securing that connection.

08 — HTTP VS HTTPS COMPARISON

The difference, side by side

FEATURE	HTTP	HTTPS
Security	No encryption	Encrypted
Data Privacy	Low	High
Uses SSL/TLS	No	Yes
Browser Padlock	No	Yes
Suitable for Login Pages	No	Yes
Suitable for Online Payments	No	Yes
SEO Advantage	No	Yes

09 — HTTP REQUEST AND RESPONSE

Every browser communication has two parts

REQUEST

Sent by the browser

— Requested URL
— Request method
— Browser details
— Cookies

RESPONSE

Sent by the server

— Status code
— HTML
— CSS
— JavaScript
— Images
— Requested data

10 — HTTP METHODS

Different verbs for different intentions

GET

Retrieve information

Example: open a webpage.

POST

Send new information

Example: submit a registration form.

PUT

Update existing information

Example: edit a user profile.

DELETE

Remove information

Example: delete a comment.

These methods are heavily used in APIs and backend development.

// GO DEEPER

The PDF **"What Is an API?"** explains how HTTP methods are used when frontend applications communicate with backend servers.

11 — HTTP STATUS CODES

Every response arrives with a verdict

200

Success — everything worked correctly.

301

Page has permanently moved.

404

Requested page was not found.

403

Access denied.

500

Internal server error.

These codes help developers identify and troubleshoot problems quickly.

12 — REAL-WORLD EXAMPLE

Logging into an online banking website

You enter your username and password. If the website used plain HTTP, someone intercepting the network traffic could potentially read this information.

// WITH HTTPS

Your credentials are encrypted before they leave your browser. Even if someone captures the data, they cannot easily understand it because it is encrypted. This is exactly why every banking website uses HTTPS.

13 — COMMON BEGINNER MISTAKES

Where most beginners get stuck

- × Thinking HTTP and HTTPS are completely different technologies.
- × Assuming HTTPS makes a website impossible to hack.
- × Believing the padlock icon means the website is trustworthy in every way.
- × Ignoring HTTPS during local development and deployment.
- × Confusing encryption with authentication.

14 — PRACTICAL ACTION PLAN

Inspect five websites' connections

- Check whether it uses HTTP or HTTPS.
- Click the padlock icon.
- View the connection information provided by your browser.

Think about why secure communication is important for each website.

15 — KEY TAKEAWAYS

What to carry forward

- HTTP is the communication protocol used by browsers and servers.
- HTTPS is the secure version of HTTP.
- HTTPS uses TLS to encrypt communication.
- Modern websites should always use HTTPS.
- HTTP methods define the type of operation being performed.
- Status codes indicate the result of every request.

// THE TAKEAWAY

If you're deploying anything that touches user data — which is nearly everything — HTTPS isn't optional. It's the baseline.

// 16 — SUMMARY PAGE

Cheat sheet

The whole guide, compressed to ten lines.

http

Enables browser and server communication

https

Encrypts communication

tls

Provides secure encryption

requests

Sent by browsers

responses

Returned by servers

get

Retrieves data

post

Sends new data

200

Means success

404

Means page not found

500

Means server error

// 17 — HTTP VS HTTPS DECISION FRAMEWORK

Which one, when?

use http

Low-stakes, local contexts only

✔ Local testing · ✔ Private internal environments

use https

Anything public or user-facing

✔ Public websites · ✔ Login systems · ✔ E-commerce websites · ✔ Banking applications · ✔ APIs · ✔ Any application handling user data

Modern production websites should always use HTTPS.

// 18 — RELATED RESOURCES

Keep going

why read it

How the Internet Works

Understand how information travels before HTTP communication begins.

why read it

What Happens When You Type a URL?

Learn where HTTP fits into the complete webpage loading process.

why read it

What Is DNS?

Understand how browsers find the correct server before sending HTTP requests.

why read it

What Is an API?

Learn how applications use HTTP methods to exchange data.

why read it

Authentication vs Authorization

Discover how secure login systems depend on HTTPS.

// 19 — RECOMMENDED NEXT LEARNING PATH

Where to go from here

1. What Is a Website?
2. Website vs Web Application
3. How the Internet Works
4. What Happens When You Type a URL?
5. **HTTP vs HTTPS — you are here**
6. What Is a Domain Name?
7. What Is DNS?

haas.dev

Engineering mindset over syntax memorization. Learn to think like a systems builder, one fundamental at a time.

haas.dev