KF4009 Web Technologies
Information security issues — part 2

The Royal Academy of Engineering funded a Visiting Professorship in Practical Cybersecurity Insights at Northumbria University, 2019–2022. These slides are a slightly modified version of those delivered, intended to be available after the project has ended

Web `https://green-pike.co.uk/nvp`
Email p.brooke@northumbria.ac.uk (until it stops working. . .)
phil@green-pike.co.uk

Previous part

- Web apps overview
- Database issues
- Simple validation and host security

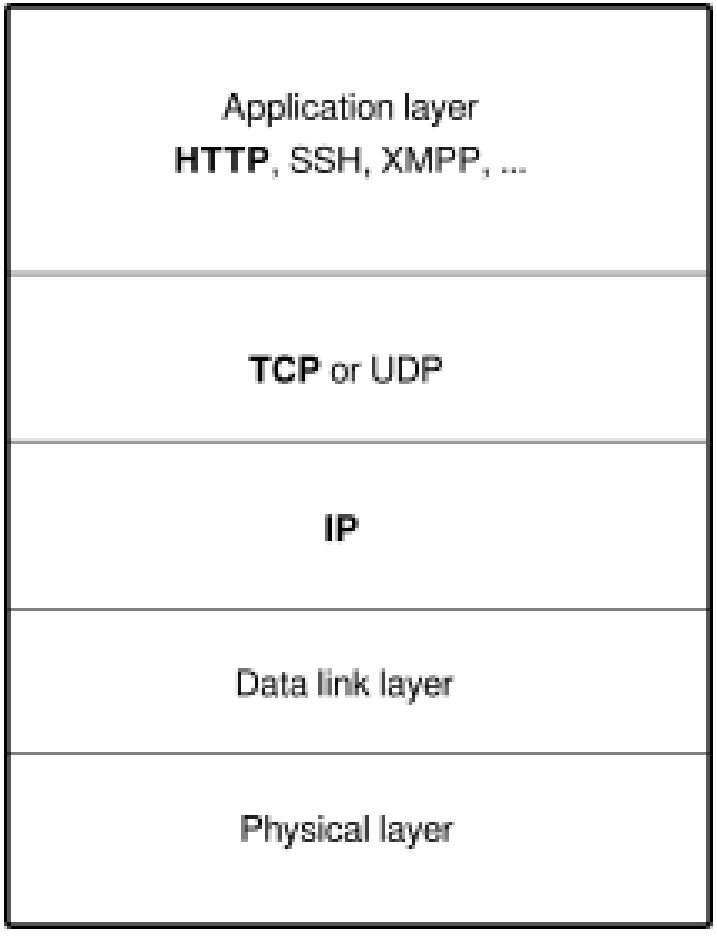
This time

- HTTP horrors
- More on validation and character sets
- Packaged web apps
- Attacker viewpoint

HTTP

Hypertext transfer protocol

- An application-layer protocol
- Sits on top of TCP
- Relatively simple (once upon a time)...



HTTP example

Client sends

```
GET / HTTP/1.0
```

Server replies

```
HTTP/1.1 200 OK
Date: Mon, 08 Feb 2010 09:51:48 GMT
Server: Apache
Accept-Ranges: bytes
Content-Length: 6446
Connection: close
Content-Type: text/html; charset=UTF-8

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html lang="en">
...
  </body>
</html>Connection closed by foreign host.
$
```

Evolution

HTTP/1.1 offers pipelining / persistent connections / "keep-alive"

HTTP/2 formerly "SPDY", broadly similar semantics with binary format. Concept of "streams" within a connection

HTTP/3 very new, *a.k.a.* HTTP-over-QUIC, binary datagram protocol...

HTTP headers

- Great for debugging problems
- Good web browsers have a console where you can watch the network transactions
- . . . but also tell me lots of useful things

It can be useful to know the web server software and version

Why?

. . . move onto the next video after you've thought about this

Encryption

- Encryption at rest
- Encryption in transit

Examples

. . . at rest	Encryption of entire databases, or perhaps only tables or columns within tables Encryption of removable media or computer hard drives
. . . in transit	Most common example: HTTPS — HTTP over SSL (more usually TLS)

Encryption is a complex subject in its own right. . . but is a critical tool for protecting data

HTTP headers

. . . can be set to enforce security properties

- HTTP Strict Transport Security (HSTS): *e.g.*,

  ```
  Strict-Transport-Security: max-age=31536000; includeSubDomains
  ```

  Directs clients to use HTTPS; if HTTPS is unavailable, to refuse the connection

- Refuse/control framing by other sites (helps defeat click-jacking)

  ```
  X-Frame-Options: DENY
  ```

  Arguably obsolete, with

  ```
  Content-Security-Policy: frame-ancestors 'none'
  ```

  an alternative

HTTP headers

- Content-Security-Policy is powerful. Example from a web app that takes payments and uses some Google fonts

```
Content-Security-Policy: default-src 'self' https://checkout.stripe.com https://js.stripe.com;
                         img-src 'self' https://*.stripe.com;
                         style-src 'self' https://fonts.googleapis.com;
                         font-src 'self' https://fonts.gstatic.com
```

Validation revisited

SQL injection

Consider a naive SQL statement, *e.g.*,

`SELECT FROM USERS WHERE '`*user input*`';`

So with a suitable user input...

```
SELECT FROM USERS WHERE ''; CREATE USER 'evil'
IDENTIFIED BY '...';
```

Prepared statements are a good mitigation...

Simple blacklisting of characters might not help...

Validation and character sets

Are you using ASCII? UTF-8? UTF-16?

What is sent to your app?

OWASP (appendix D of the testing guide):

- "For instance, a / can be represented as 2F (hex) in ASCII, while the same character (/) is encoded as C0 AF in Unicode (2 byte sequence)."
- "Multibyte encoding has been used in the past to bypass standard input validation functions and carry out cross site scripting and SQL injection attacks."
- Misuse of numeric encodings (*e.g.*, hex)

Validation and character sets

Many programming languages are not directly aware of multibyte encoding

- C strings are simply a sequence of octets, with the variable holding the starting memory address

PHP has a set of multibyte string functions:
`https://www.php.net/manual/en/ref.mbstring.php`

> *"Multibyte character encoding schemes and their related issues are fairly complicated"*

Validation and character sets

Suggestions for further reading:

- The Open Web Application Security Project (OWASP) `https://www.owasp.org/` is an excellent resource
- Internationalized Domain Names in Applications (IDNA)
- Punycode

Aside: As well as allowing *many* more ASCII domain names, ICANN also allows UTF-8 domain names... beware of homograph attacks *a.k.a.* script spoofing

More details at, for example: `https://blog.malwarebytes.com/101/2017/10/out-of-character-homograph-attacks-explained/`

Electron

Embeds a Chromium web renderer and the Node JavaScript engine

- Easy to take an existing web app and re-deploy it as a "desktop" application

Examples:

- Signal messenger
- Riot (a Matrix client)
- Discord (text/video gamers chat client)
- Teams

What problems does this bring?

. . . move onto the next video after you've thought about this

Attacker's mindset

Example: I want the web app's database

- Is the site using a known-bad version of an application? (*e.g.*, Wordpress)
- Is there a broken or vulnerable web server?
- Does it have a naively written web app? Run a fuzzer against it...
- Is the firewall set up badly? Are unnecessary services visible? (Can I exploit them?)
- Can I use social engineering to get a privileged password?

The end

Web	`https://green-pike.co.uk/nvp`
Email	p.brooke@northumbria.ac.uk (until it stops working. . .)
	phil@green-pike.co.uk

Phil Brooke
A *Royal Academy of Engineering Visiting Professor* in Practical Cybersecurity Insights
Northumbria University