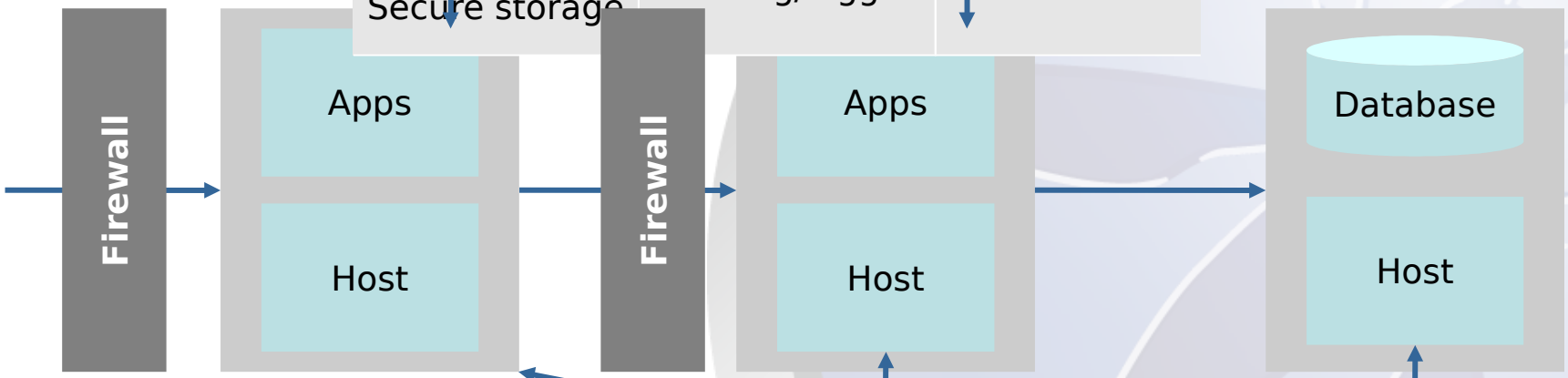
Web Application Security

The OWASP Foundation
http://www.owasp.org

Securing the application		
Input validation	Session mgmt	Authentication
Authorization	Config mgmt	Error handling
Secure storage	Auditing/logging	



Securing the network
Router
Firewall
Switch

Securing the host		
Patches/updates	Accounts	Ports
Services	Files/directories	Registry
Protocols	Shares	Auditing/logging

Web Application Behaviour

- HTTP is stateless and hence requests and responses to communicate between browser and server have no memory.
- Most typical HTTP requests utilise either GET or POST methods
- Scripting can occur on:
  - Server-Side (e.g. perl, asp, jsp)
  - Client-Side (javascript, flash, applets)
- Web server file mappings allow the web server to handle certain file types using specific handlers (ASP, ASP.NET, Java, JSP,CFM etc)
- Data is posted to the application through HTTP methods, this data is processed by the relevant script and result returned to the user's browser

HTTP POST
HTTP GET

"GET" exposes sensitive authentication information in the URL

- In Web Server and Proxy Server logs
- In the http referer header
- In Bookmarks/Favorites often emailed to others

"POST" places information in the body of the request and not the URL

Enforce HTTPS POST For Sensitive Data Transport

GET vs POST HTTP Request

GET request

GET /search.jsp?
name=blah&type=1
HTTP/1.0
User-Agent: Mozilla/4.0
Host: www.mywebsite.com
Cookie:
SESSIONID=2KDSU72H9GSA
289
<CRLF>

POST request

POST /search.jsp HTTP/1.0
User-Agent: Mozilla/4.0
Host: www.mywebsite.com
Content-Length: 16
Cookie:
SESSIONID=2KDSU72H9GSA
289
<CRLF>
name=blah&type=1
<CRLF>

What are HTTP Headers?

HTTP headers are components of the message header of HTTP Requests and Responses
HTTP headers define different aspects of an HTTP transaction

HTTP headers are colon-separated name-value pairs in clear-text string format, terminated by a carriage return (CR) and line feed (LF) character sequence.

http://en.wikipedia.org/wiki/List_of_HTTP_header_fields

Security HTTP Response Headers

- X-Frame-Options
- X-Xss-Protection
- X-Content-Type-Options
- Content Security Policy
- Access-Control-Allow-Origin
- HTTPS Strict Transport Security
- Cache-Control / Pragma

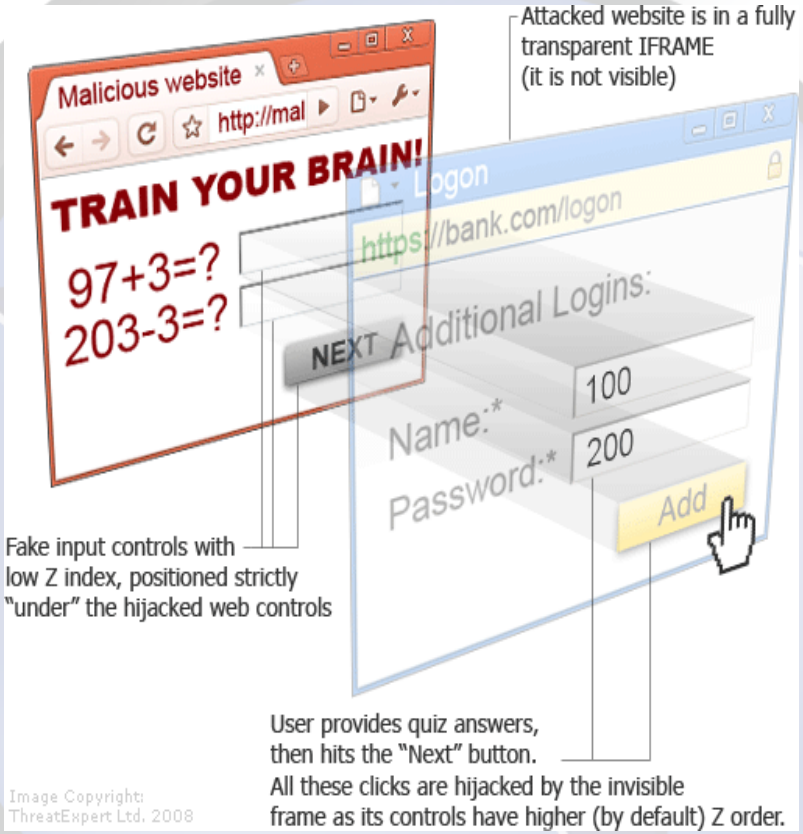
Security HTTP Response headers

X-Frame-Options '*SAMEORIGIN*' - allow framing on same domain. Set it to 'DENY' to deny framing at all or 'ALLOWALL' if you want to allow framing for all website.
X-XSS-Protection '*1; mode=block*' - use XSS Auditor and block page if XSS attack is detected. Set it to '0;' if you want to switch XSS Auditor off(useful if response contents scripts from request parameters)
X-Content-Type-Options '*nosniff'* - stops the browser from guessing the MIME type of a file.
X-Content-Security-Policy - A powerful mechanism for controlling which sites certain content types can be loaded from
Access-Control-Allow-Origin - used to control which sites are allowed to bypass same origin policies and send cross-origin requests.
Strict-Transport-Security - used to control if the browser is allowed to only access a site over a secure connection
Cache-Control - used to control mandatory content caching rules

X-Frame-Options

Protects you from most classes of Clickjacking

X-Frame-Options: DENY
X-Frame-Options: SAMEORIGIN
X-Frame-Options: ALLOW FROM



X-XSS-Protection

Use the browser's built in XSS Auditor

X-XSS-Protection: [0-1](; mode=block)?

X-XSS-Protection: 1; mode=block

X-Content Type-Options

Fixes mime sniffing attacks

Only applies to IE, because only IE would do something like this

X-Content-Type-Options = ‘nosniff’

Content Security Policy

- Anti-XSS W3C standard http://www.w3.org/TR/CSP/
- Move all inline script and style into external files
- Add the X-Content-Security-Policy response header to instruct the browser that CSP is in use
- Define a policy for the site regarding loading of content
- Chrome version 25 and later (50%)
- Firefox version 23 and later (30%)
- Internet Explorer version 10 and later (10%)

Strict Transport Security

Strict-transport-security: max-age=10000000

Do all of your subdomains support SSL?

Strict-transport-security: max-age=10000000; includeSubdomains

Disabling the Browser Cache

Add the following as part of your HTTP Response

Cache-Control: no-store, no-cache, must-revalidate
Expires: -1

HTTP Security Headers Tool

Secure headers!
Open source
https://github.com/twitter/secureheaders