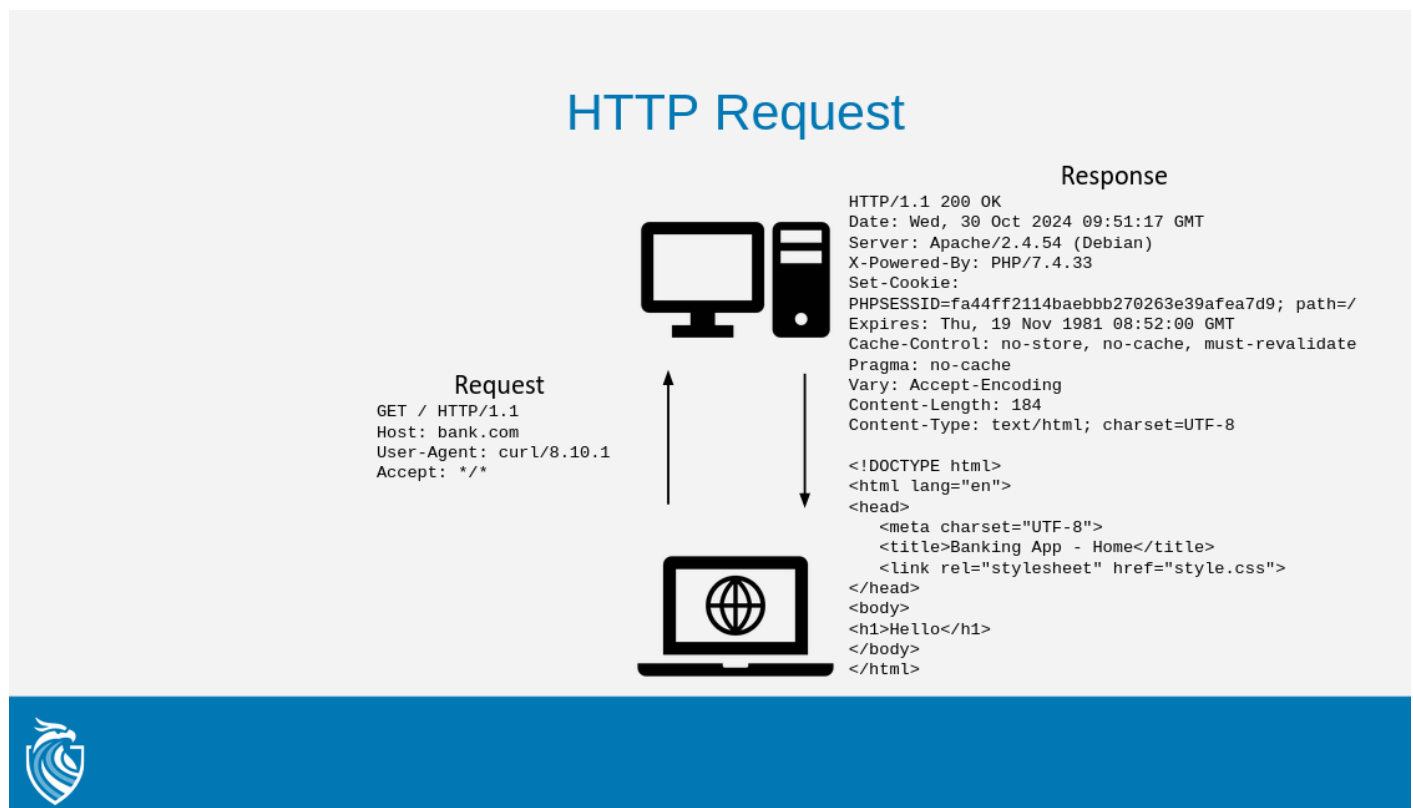


Introduction to web

The slides can be found [here](#), technical files for the workshop can be found [here](#)

The modern web

When you open up your browser and go to a website, what happens? Well, after a bunch of networking steps are taken your computer sends a HTTP request to the website, specifically the web server. This server is a different computer that is running the website. The web server then processes the HTTP request and sends back an HTTP response, the overall interaction looks like this:



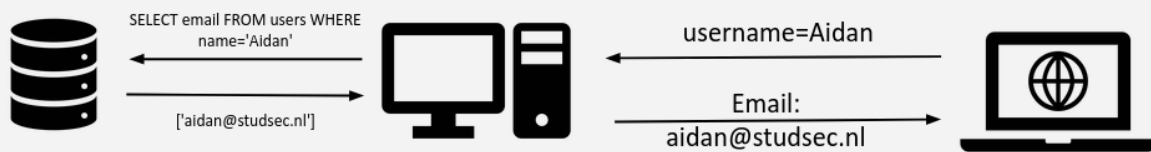
The first part is called the HTTP headers, these are key value pairs ending on a newline. After the newline comes the optional HTTP body, which can be seen in the response sent back by the server.

However, nowadays the interaction is often not as simple as that. The web server in turn might interact with other servers to process the HTTP request. One of these servers might be a database.

A simple search engine

Suppose our website implements a search feature, that allows users to search entries in a database. The web server processes the users HTTP request and then uses it to query the database. The result is then sent back to the user, the overall interaction will look like:

Database interactions



Here you will notice the request sent to the database from the web server, "SELECT email FROM users WHERE name='Aidan'", this is an SQL query that will return all users who's name is "Aidan". But what happens when we add an apostrophe in our username query? And what if we then add some SQL after that? The result can be seen in the following image, note that variables are highlighted in red.

Database interactions

Username=Aidan' OR '1'='1

PHP: SELECT email FROM users WHERE name='Aidan' OR '1'='1'

SQL: SELECT email FROM users WHERE name='Aidan' OR '1'='1'

What will this return?



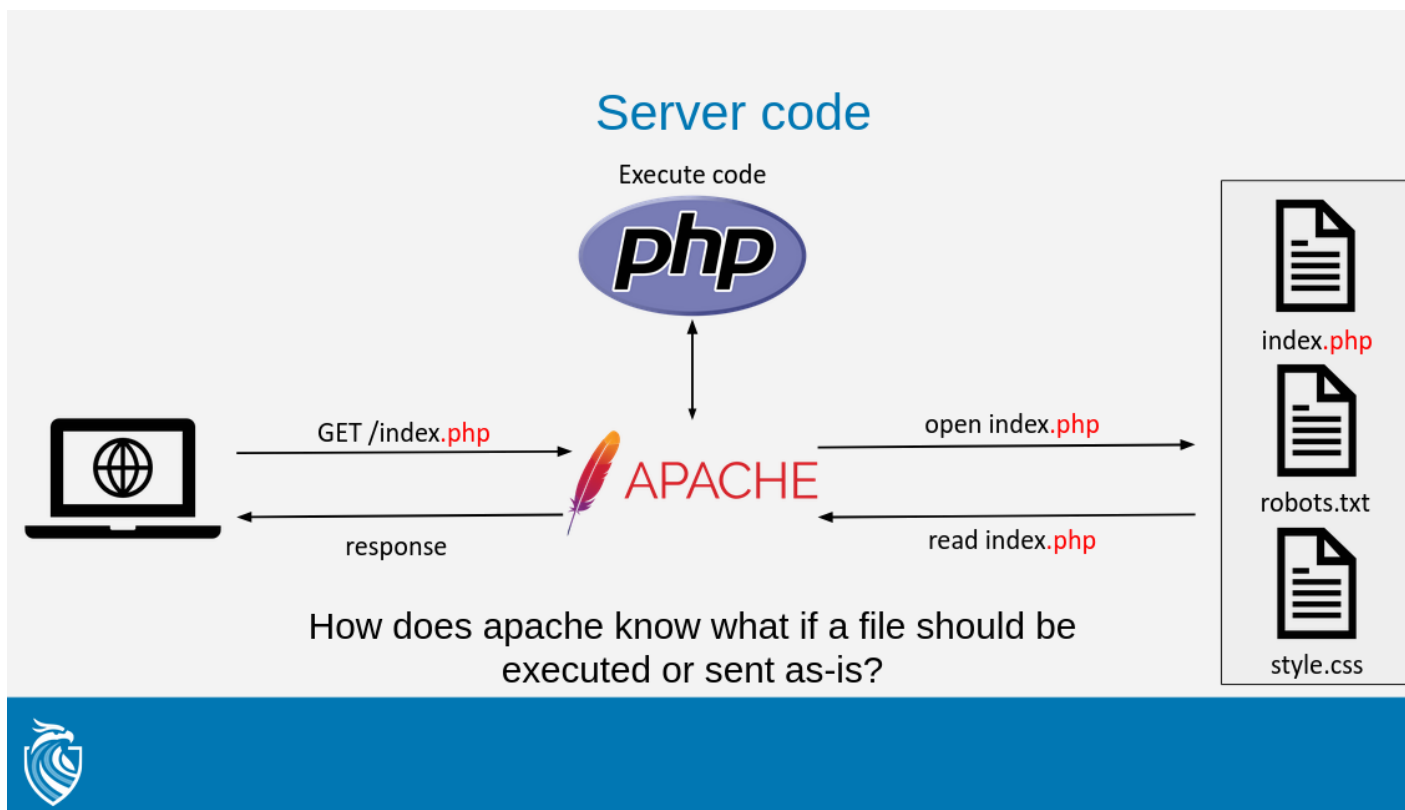
Here the web server (PHP) is asking for the email address of all users whose name is "Aidan' OR '1'='1". However, SQL will interpret the same string as "Give me the email of all users where the name is 'Aidan' or all users where 1=1", this will always be the case, meaning the database will return all entries.

Why is this a problem, we are supposed to query the database right? Correct, but we are only allowed to get the "email" field of all users, by adding our own SQL we can get *all* information in the database, including possibly sensitive information like passwords.

Uploading a file

Lets now turn our attention to the web server itself. Originally most websites were based on a folder structure, where each file could be accessed as a web page. This worked brilliantly for simply sharing and displaying files, where code (if any) would be run on the computer accessing the files. However, more complex systems, such as logging a user in or searching a database also required code to run on the web server.

One way this was implemented is with PHP, existing files could define PHP code that would be run on the server when the page was accessed. To distinguish between code that was intended to run on the server and normal text the web server looks at the file extension, a ".php" file will first execute the PHP code and then send the result to the client.



So far so good, but what if a website allows you to upload files? Say a profile picture or as personal storage, how does the web server know the difference between the original files and files that a user uploaded? Well, it doesn't, if a user manages to upload a file with a ".php" extension then they are able to run PHP code on the web server. This is a big problem, as it allows a user to run any command on the server as if it were their own computer, essentially giving them full control over the web server.

Revision #1

Created 20 November 2024 18:59:03 by delta6862

Updated 20 November 2024 19:58:07 by delta6862