

# Discussion Week 5

Crypto and PA4

# Logistics + Agenda

PA3 due tomorrow at 11AM PST

Advice: don't hard-code values, even if they work on your machine!

Both memory organization and timing vary from machine to machine.

Today:

Review of Crypto + PKI (Hannah)

Introduction of PA4 and Network Security (Patrick)

# Reviewing Crypto Primitives and their Goals

## Confidentiality

Symmetric/Asymmetric encryption

## Confidentiality + Integrity

## Integrity/Authenticity

Message Authentication Codes

Digital Signatures

Authenticated encryption

TLS

## Identity

Public-Key Infrastructure

## A challenge:

You've just started a new job at a start-up that publishes a blog. Your company is building an app that allows users to download specific pages of the blog on demand. What type of crypto should you use to secure this feature?

# Solution:

## Features needed:

**Confidentiality:** Although the blog is public, communication should not reveal which pages a user requested.

**Integrity:** The user should receive the exact, unaltered pages of the blog they request.

**Identity of the server:** The user should be able to trust that no one but the blog server can decrypt or respond to the request:

**TLS + PKI!** TLS contains authenticated encryption for confidentiality and integrity; PKI ensures that no man-in-the-middle attacks are occurring.

## Challenge 2:

You are building a mirror of several large, open-source projects. You would like to prove to users that the files downloaded from the mirror are identical to those at the original host. What crypto primitive could you use to do this efficiently?

# Solution

Features required:

No one should be able to create “proof” that the files match if they do not.

-Integrity? Give the original host a private key, let them sign the original files. Then distribute the signature with duplicates.

-Works, BUT: Should the mirror give out a certificate for host? What happens when the certificate is revoked or expired? Need ongoing cooperation of original host.

-Better: Two distinct outputs will not generate the same “proof”.

**Collision resistance: SHA2 or SHA3**

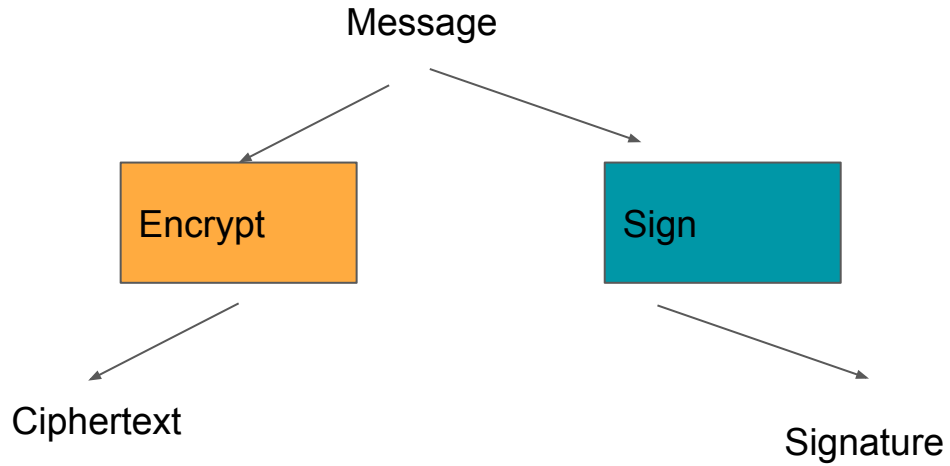
-Original host computes SHA256(files) and posts this on the original server.

-When users download mirrored files, they compute SHA256(files) and compare.

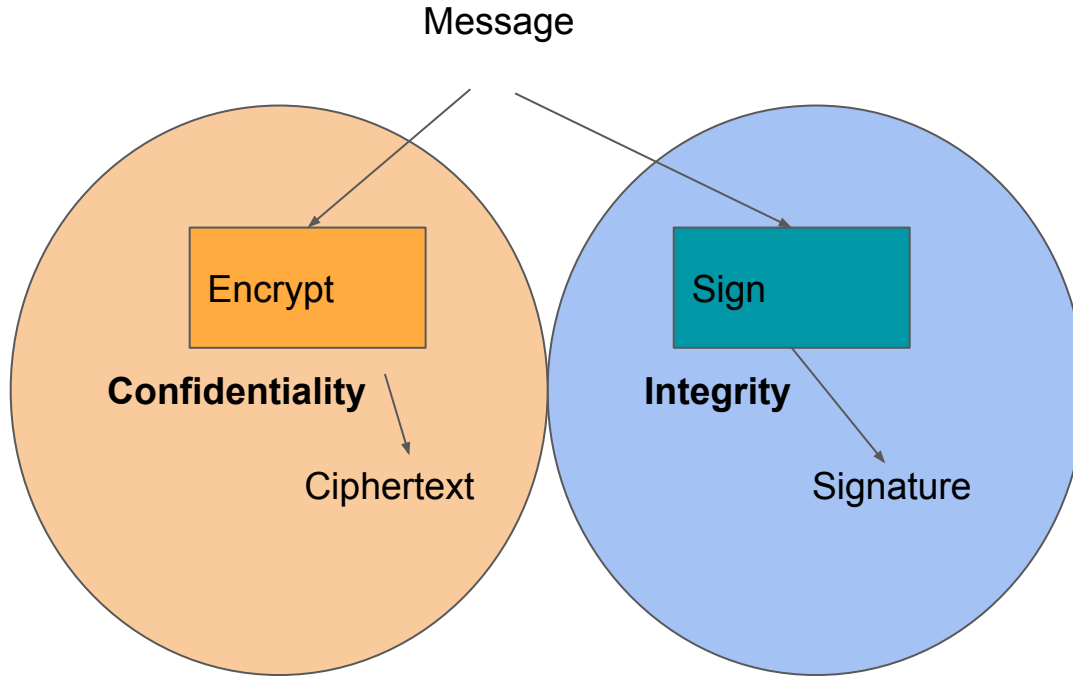
# Why don't we roll our own crypto?

- Assume we encrypt and sign a message from Alice to Bob  
Assume decryption is successful and the signature verifies
- What can Alice and Bob assume?
  
- Bob knows that
  - Alice knows the plaintext (and anyone else she shared it with or copied it from)
  - Alice (or someone with her private key) signed the plaintext at some point in the past
- Alice knows that
  - Only Bob (or someone with his private key) can extract the plaintext from the encrypted channel
  - Bob (or anyone else) can prove that Alice (or someone with her private key) signed the plaintext

# Why don't we roll our own crypto?



# Why don't we roll our own crypto?



Remember:

Only Bob (or someone who knows his private key) can extract the plaintext from the **encrypted** channel.

But **anyone** can extract the plaintext from the signature!

# So how do we use crypto?

Find one existing protocol that provides all of the security features you need.

Confidentiality only: Symmetric encryption like AES-CBC or AES-CTR.

May combine with asymmetric encryption like RSA-OEAP

Integrity only: Digital signatures (asymmetric) and MACs

Confidentiality + integrity: Authenticated encryption like AES-GCM or ChaCha20+Poly1305

Collision resistance: cryptographic hash function like SHA2 or SHA3

Randomness: Cryptographically secure random number generator

Secure channel: TLS

# Crypto Review: Public Key Infrastructure

Asymmetric primitives are only effective when Bob can be certain he knows Alice's key.

Two ways for this to happen in practice:

- Decentralized (PGP, Keybase): Follows social network of trust.

- Centralized: Certificate authorities

# Web of Trust: what actually happens at a key-signing party

Here's my public key:

Please sign a document saying that this public key belongs to Hannah Davis, who owns the email [h3davis@ucsd.edu](mailto:h3davis@ucsd.edu).

What do you trust?

- Is this actually my public key (who has access to the Zoom meeting?)
- Am I Hannah Davis?
- Am I reporting my email honestly?

Should your friends trust your word on all of this information? What about their friends?

The advantage: Everyone gets to set their own level of comfort.

The disadvantage: How do you explain this to your grandmother?

# Why don't we use PKI in email?

You can, with PGP:

-----BEGIN PGP SIGNATURE-----

Comment: GPGTools - <http://gpgtools.org>

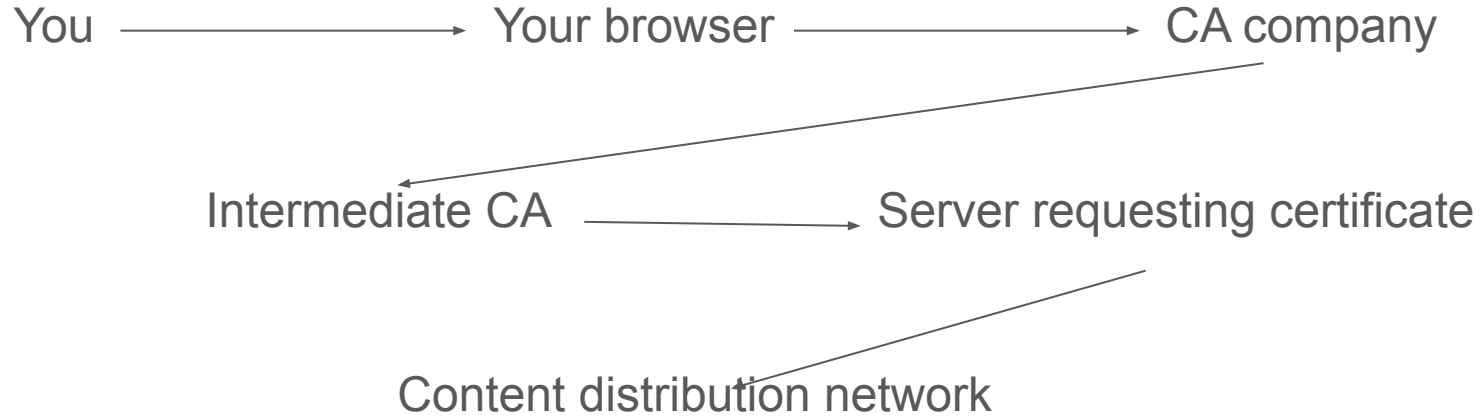
```
iQJRBAEBCAA7FIEEyRg3TFk4qfnEtSUUStX143RDyIQFAMb1SwYdHHRpYm9yLmph
Z2VyQHVuaS13dXBwZXJ0YWwuzGUACgkQStX143RDyIT7KQ/+LFEG33wYuPQlxBVT
oYJEUg2WdA0R4GyQI+y8QYhSFhgCvb6ZcypMI/Fi7st23hJi5PNjqJGpveXZ63+0
Szl4Hf4L/oRBXiuOqwLqjXwALgtAG7RnAI8MX5XmWx.Jx88Mqgdpb5s8hAX5Q03uJ
A/OOZt5npFyxd5PmB0hxPa6JpeDyyHh3linnAZqrFSc02y8LQi8Vmhe0WRc2L1e
9BFj0BmMmzNT8F6ac2clQwV8EWJn7pzoMokF4fnsRH9T88qCQd31JJmo+D2e9amL
f6wGeBqurxgN/0/QKMn+obKWDq/I531jyUv2+M5/i1E6sKXuPwbQhHm3RknNr/t9
Tk4zDHxhw2jxpuH7URH8S3Vl/9aLojNNDVdZ5E175rKXc3eEoivoRUuUNQoCoyyF
/B2kl+yeRViiW/TWfbL4hULffgabFSBltGwz+YKOEoos5wd1oe7tBJng1HheixU5
N3BTn4g2mf9zfXXpeB6G2JD+biVzHchb3fX8x79zZPjPrRczBIHAT5lwFnzU6MOB
vLayFpneOmtVw9eU6H+rMMZrDa3km+Emhqv0qR1BRsgd7iyCu6kMPI8yV/rsSM7Z
xM2a688LZBviXu4L7lgGIOVA1hTGbZuXeBoGGV2CC59Xj0gKXsiU6XWtfYlwjb2X
nY3yQHvi3L9SQv/QvOf28GTGTZ0=
=qx2P
```

-----END PGP SIGNATURE-----

But most commercial email programs do not support this (thanks to ad \$\$\$).

# Certificate Authorities

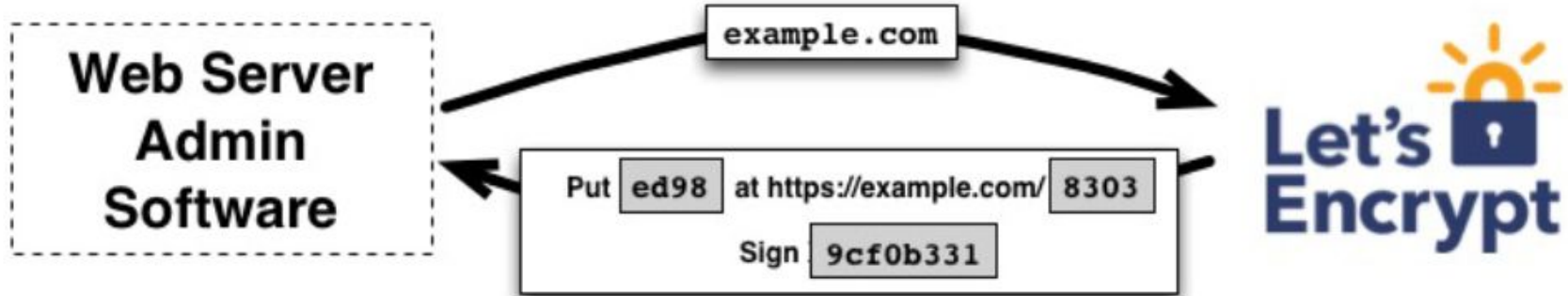
Chain of trust:



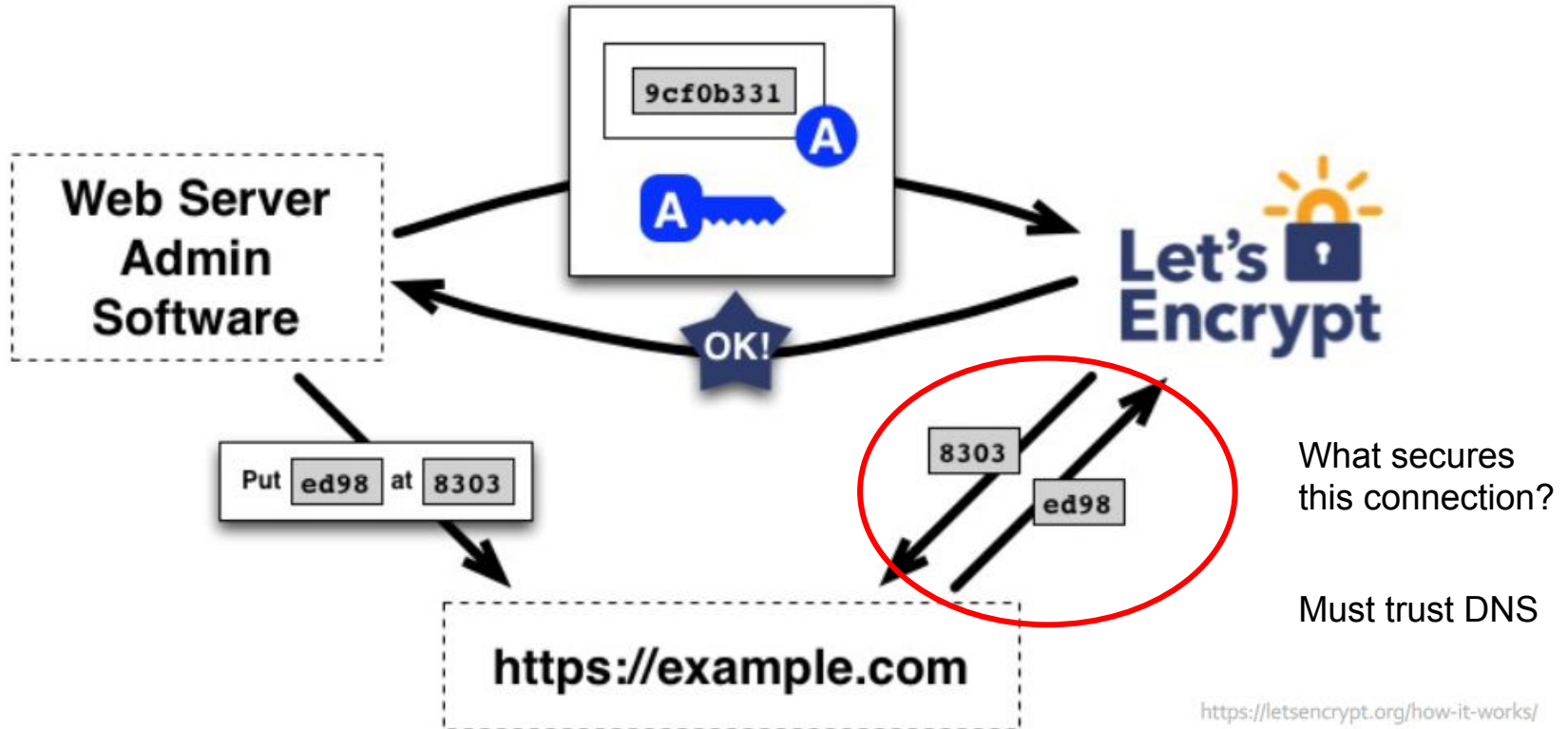
Signature verification works back up this chain!

# How to obtain a certificate

Demonstrate (legal) ownership of a domain OR demonstrate **control** of a domain.



# Let's Encrypt Certificate generation



Pause for questions about crypto



# PA4

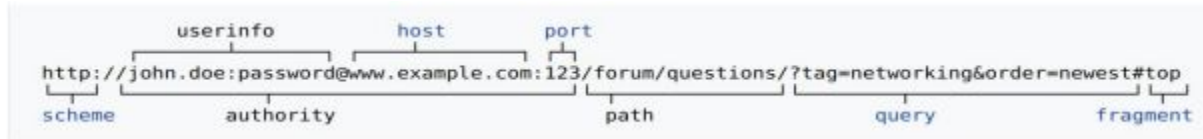
Slides adapted from Riley Hadden

# PA4 Topics

- Web Security
- HTTP/HTTPS
  - GET, POST, etc.
- JavaScript injection
  - Base64 encoding
  - SQL injection

# HTTP/HTTPS

- HTTP: HyperText Transfer Protocol
- HTTPS = HTTP + SSL/TLS
- Common HTTP Methods
  - GET
  - POST
  - PUT
  - DELETE



# Using Python

- GET

```
import requests

# Optional parameters for GET request
params = {}
url = "https://www.google.com"
response = requests.get(url, params)

# Generally
# 200 -> all good
# 404 -> resource not found
# 500 -> error on server side
response.status_code

# This will have the response body
response.content
```

- POST

```
import requests

# This will be the data the sender requests the server to store/update
data = {"username": "riley",
        "password": "youllneverguessit!",
        "favorite color": "blue"}

url = "https://www.google.com"
response = requests.post(url, data)
```

# Web Page Anatomy(simplified)

- HTML
  - Contains layout of page
- CSS
  - Allows for formatting of pages in a consistent style
- Javascript
  - Allows dynamic content
  - Web browser runs code locally

# Visiting a Webpage

Steps:

1. Browser requests a webpage
2. Server sends back resources(HTML, CSS, Javascript)
3. Browser uses resources to render page
4. Browser runs embedded Javascript
  - a. Supports interactive features like responses to clicking, infinite scroll, and more
  - b. Can load larger resources later

# Using the Developer Tools(Console)

The image displays a web browser window with the w3schools.com website on the left and the Chrome Developer Tools console on the right. The website content includes:

- HTML**: The language for building web pages. Buttons: LEARN HTML, HTML REFERENCE.
- CSS**: The language for styling web pages. Buttons: LEARN CSS, CSS REFERENCE.
- JavaScript**: The language for programming web pages. Button: LEARN JAVASCRIPT, JAVASCRIPT REFERENCE.

The Developer Tools console shows the following content:

- Network tab: A list of requests, including several blocked by the client and some from adtech services.
- Warnings tab: A list of DOM warnings for missing elements and size mismatches.
- Console tab: A highlighted JavaScript object: `<> HTMLCollection(7) [img, goog-te-gadget-icon, img, img, img, img, img]`. Below it, a `Location` object is shown with `href: "https://www.w3schools.com/"`.
- Help tab: A sidebar with "Highlights from the Chrome 70 update", "Live Expressions in the Console", "Highlight DOM nodes during Eager Evaluation", and "Autocomplete Conditional Breakpoints".

# Using the Developer Tools(Elements)

The image shows a browser window displaying the w3schools.com website. The page content includes:

- Logo: w3schools.com
- Navigation: HTML, CSS, JavaScript
- Sub-headers: "The language for building web pages", "The language for styling web pages", "The language for programming web pages"
- Buttons: LEARN HTML, HTML REFERENCE, LEARN CSS, CSS REFERENCE, LEARN JAVASCRIPT, JAVASCRIPT REFERENCE

The Chrome Developer Tools are open, showing the following panels:

- Elements:** Displays the HTML DOM tree. The selected element is `<h1>HTML</h1>` with the following structure:

```
<!doctype html>
<html lang="en-US" style="height: 100%;">
  <head>...</head>
  <body style="position: relative; min-height: 100%; top: 0px; class=" darktheme">
    <div class="w3-row w3-white w3-padding w3-hide-medium w3-hide-small" style="margin-top:5px;">...</div>
    <div class="w3-hide-large w3-hide-large w3-padding-16">...</div>
    <div style="display:none;position:absolute;z-index:6;right:60px;height:57px;padding-top:12px;padding-right:20px;background-color:#4CAF50;" id="googleSearch">...</div>
    <div style="display:none;position:absolute;z-index:5;right:120px;height:57px;background-color:#4CAF50;text-align:right;padding-top:15px;" id="google_translate_element">...</div>
    <div class="w3-bar w3-theme w3-card-2 w3-wide notranslate">...</div>
    <nav class="w3-sidebar w3-collapse w3-white w3-card-2" id="mysidenav">...</nav>
    <div id="myAccordion" class="w3-card-2 w3-light-grey w3-center w3-hide-large" style="display:none;cursor:default">...</div>
    <nav id="nav_tutorials" class="w3-light-grey w3-card-2 w3-hide-small nav">...</nav>
    <nav id="nav_references" class="w3-light-grey w3-card-2 w3-hide-small nav">...</nav>
    <nav id="nav_examples" class="w3-light-grey w3-card-2 w3-hide-small nav">...</nav>
    <nav id="nav_exercises" class="w3-light-grey w3-card-2 w3-hide-small w3-hide-medium nav">...</nav>
    <!-- MAIN -->
    <div class="w3-main" id="main">
      <div class="w3-row w3-margin-bottom">
        <div class="w3-col l6 w3-center" style="padding:3%">
          <h1>HTML</h1>
          <p class="w3-xlarge w3-text-dark-grey">The language for building web pages</p>
          <a href="/html/default.asp" class="w3-button w3-dark-grey">LEARN HTML</a>
          <a href="/css/default.asp" class="w3-button w3-dark-grey">HTML REFERENCE</a>
        </div>
        <div class="w3-col l6" style="padding:3%">...</div>
      </div>
      <div class="w3-row w3-light-grey w3-hide-medium w3-hide-small">...</div>
      <div class="w3-row w3-light-grey w3-hide-large" style="padding-bottom:30px">...</div>
      <div class="w3-row">...</div>
      <div class="w3-row w3-light-grey w3-hide-medium w3-hide-small">...</div>
      <div class="w3-row w3-light-grey w3-hide-large" style="padding-bottom:30px;margin-top:40px">...</div>
      <div class="w3-row w3-dark-grey w3-padding-32">...</div>
      <div class="w3-row w3-white" style="padding-top:40px;padding-bottom:40px">...</div>
      <div class="w3-dark-grey w3-center w3-padding-large w3-padding-48" style="padding-bottom:68px!important">...</div>
      <div class="w3-light-grey w3-center w3-padding-large w3-padding-32">...</div>
      <div class="w3-white w3-center w3-padding-large w3-padding-16" style="padding-bottom:36px!important">...</div>
      <div class="w3-display-container w3-black" id="certsection">...</div>
      <footer class="w3-container w3-dark-grey w3-center w3-padding 32">...</footer>
    </div>
  </div>
  <!-- END MAIN -->
</html>
```
- Styles:** Shows the default user agent styles for the `h1` element, such as `font-size: 2em`, `font-weight: bold`, and `margin-bottom: 0.67em`.
- Console:** Displays several error messages from the browser's console, all with the message: `[s]nbh](3.0445): Missing DOM element 'snhb-main_leaderboard-0' for auctioned ad-unit. Skipping in ad server refresh.`

# Using the Developer Tools(Sources)

The image shows a web browser displaying the w3schools.com homepage. The page features a green header with the w3schools.com logo and a search icon. Below the header, there are three main sections: HTML, CSS, and JavaScript, each with a large heading and a sub-heading. The HTML section is titled "HTML" and "The language for building web pages", with links for "LEARN HTML" and "HTML REFERENCE". The CSS section is titled "CSS" and "The language for styling web pages", with links for "LEARN CSS" and "CSS REFERENCE". The JavaScript section is titled "JavaScript" and "The language for programming web pages", with links for "LEARN JAVASCRIPT" and "JAVASCRIPT REFERENCE".

The developer tools are open to the Sources tab, showing the file system and the source code of the page. The file system shows a directory structure with files like w3css, w3codecolor.js, (index), getcertified.jpg, and w3css\_templates.jpg. The source code is displayed in a dark theme, showing HTML, CSS, and JavaScript code. The JavaScript code includes functions for setting the logo, privacy policy, and locale. The console shows a "What's New" message from Chrome 70, highlighting new features like Live Expressions in the Console, Highlight DOM nodes during Eager Evaluation, and Autocomplete Conditional Breakpoints.

# Using the Developer Tools(Network)

The screenshot displays the w3schools.com website with sections for HTML, CSS, and JavaScript. The Chrome DevTools Network tab is open, showing a list of requests. The following table summarizes the visible requests:

Name	Status	Type
prebid	(blocked:other)	xhr
bid?src=prebid_prebid_3_14_0	(blocked:other)	xhr
prebid	(blocked:other)	xhr
sygnus?e=277108&v=7.2&r=%7B%22id%22%3A%22483b971e1_00y8z...	(blocked:other)	xhr
bidRequest?dcm=8a9694b8017070f8309df8ed25b10036&po..._is_leader	(blocked:other)	xhr
bidRequest?dcm=8a9694b8017070f8309df8ed25b10036&po..._ols_wide...	200	xhr
bidRequest?dcm=8a9694b8017070f8309df8ed25b10036&po..._ools_mid...	(blocked:other)	xhr
bidRequest?dcm=8a9694b8017070f8309df8ed25b10036&po..._s_botto...	200	xhr
ADTECH?v=2&cmd=bid&cors=yes&alias=67462c-b0e0b41d&misc=158871...	(blocked:other)	xhr
ADTECH?v=2&cmd=bid&cors=yes&alias=668928a37a0c61f&misc=158871...	(blocked:other)	xhr
ADTECH?v=2&cmd=bid&cors=yes&alias=6902fbb4619806&misc=158871...	(blocked:other)	xhr
flowad.js	(blocked:other)	document
async_usersync.html	(blocked:other)	document
async_usersync.html	(blocked:other)	document
gd?gdr=0&gdr_consent=BoY8z-QOy8z-QDIDKAENDD-AAAAuAAA&u...	(blocked:other)	document
async.html?gdr=0&gdr_consent=BoY8z-QOy8z-QDIDKAENDD-AAAAu...	(blocked:other)	document
ismatch.html	(blocked:other)	document
async_usersync.html	(blocked:other)	document
connectmoyusers.png?gdr_consent=BoY8z-QOy8z-QDIDKAENDD-AAAA...	(blocked:other)	document
data:image/svg+xml...	200	svg+xml
data:image/svg+xml...	200	svg+xml
data:image/svg+xml...	200	svg+xml
data:image/svg+xml...	200	svg+xml
data:image/png;base...	200	png
data:image/png;base...	200	png
data:image/png;base...	200	png

The Network tab also features a waterfall chart on the right, showing the timing of these requests. The console at the bottom displays a message: "Highlights from the Chrome 70 update".

# Using the Developer Tools(Application)

The screenshot displays the w3schools.com website on the left and the Chrome DevTools Application tab on the right. The Application tab is focused on the Cookies section, which is highlighted with a red box. The Cookies table lists various cookies with columns for Name, Value, Domain, Path, Expires, Size, HTTP, Secure, and SameSite. The console at the bottom shows 'What's New' highlights from the Chrome 70 update.

Name	Value	Domain	Path	Expires / ...	Size	HTTP	Secure	SameSite
1P_JAR	2020-05-05-22	.google.c...	/	2020-06-0...	19		✓	
APISID	D9FW0DySeYPCm07Z/Aav-zv3ZH7gQJq4d	.google.c...	/	2022-05-0...	40			
DSID	AAO-7r6aMZEISLmroFV77cl-IHkAP-kTdK3pkqXSRdPI8Uwx6...	.doublecli...	/	2020-05-1...	123	✓	✓	
HSID	AcW0Vui4vdVdg7Jno	.google.c...	/	2022-05-0...	21	✓		
IDE	AHWqTUmZecgSsLmem-pGN7PcGXrnf8GnFAZI2wGU69J...	.doublecli...	/	2021-08-2...	67	✓	✓	
NID	203-w8Q_RUP94LbgvfrjCj3z_OYnRPh-KGWkOzUHVqLD9DeF...	.google.c...	/	2020-11-0...	271	✓	✓	
OGPC	19008563-30	.google.c...	/	2020-05-1...	16			
OTZ	5440246_84_88_104280_84_446940	www.goo...	/	2020-06-0...	33		✓	
S	billing-ui-v3-n8tp-UxzT1P3okIEACEnnXvVF0eToSak-billing-ui-v...	.google.c...	/	1969-12-3...	98	✓	✓	
SAPISID	J5vVWn-goNz_E3Fy/AFxwTS9auVCRSLCRK	.google.c...	/	2022-05-0...	41	✓	✓	
SEARCH_SAMESITE	CgQI048B	.google.c...	/	2020-11-0...	23			Strict
SID	wg-c3Qgwzm7xEKQs4BBJi5ivhni3i67fpAwZEMDwlSZWO73JT...	.google.c...	/	2022-05-0...	74	✓	✓	
SIDCC	AJi4QIGIhwaNISCKFE7IVLhuMan7mkioZKFz05iUah63iK3dDfE...	.google.c...	/	2021-05-0...	80		✓	
SSID	Avsh1185rd3apq1	.google.c...	/	2022-05-0...	21	✓	✓	
__Secure-3PAPISID	J5vVWn-goNz_E3Fy/AFxwTS9auVCRSLCRK	.google.c...	/	2022-05-0...	51		✓	
__Secure-3PSID	wg-c3Qgwzm7xEKQs4BBJi5ivhni3i67fpAwZEMDwlSZWO73J...	.google.c...	/	2022-05-0...	85	✓	✓	
__Secure-APISID	D9FW0DySeYPCm07Z/Aav-zv3ZH7gQJq4d	.google.c...	/	2020-08-0...	49		✓	
__Secure-HSID	AcW0Vui4vdVdg7Jno	.google.c...	/	2020-08-0...	30	✓	✓	
__Secure-SSID	Avsh1185rd3apq1	.google.c...	/	2020-08-0...	30	✓	✓	
_ga	GA1.2.638308091.1543802132	.w3school...	/	2022-05-0...	29		✓	
_gid	GA1.2.2134237268.1588718784	.w3school...	/	2020-05-0...	31		✓	
_pubcid	4f77d330-020c-423e-9b6a-0d6276bc45cc	www.w3s...	/	2024-10-1...	43		✓	
cto_bidid	XB-4dF9pWwG2YU92czVXbEp3WJJSZjNkc1b0BJRFVWNTZL...	www.w3s...	/	2021-05-3...	168		✓	
cto_bundle	STVtrV82NHV2YONJJTGaUhw1NXBIUk40UtdidWlMkl5duHjR...	www.w3s...	/	2021-05-3...	244		✓	
cto_test_cookie	1	www.w3s...	/	1969-12-3...	16		✓	

# Javascript Injection

## Source

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph with some user info: {}</p>
```

- Suppose {} takes user input
  - What happens if we supply Javascript?

## Rendering

### **This is a Heading**

This is a paragraph with some user info: {}

## Source

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph with some user info: <script>document.write("\tHello, from inside the
script!")</script></p>
```

## Rendering

### **This is a Heading**

This is a paragraph with some user info: Hello, from inside the script!

# PA4 details

- This PA is in CTF-style
  - Login credentials will be on Gradescope tomorrow, May 6
- There are 8 levels, each will have a clue
- Do not use brute-force approaches!
  - Server has a rate-limit, will ban you for the day if you exceed it
- You may find scripting helpful for some levels
  - Your scripts should be very short(<10 lines)
- The remainder of the concepts required for this PA will be covered next week