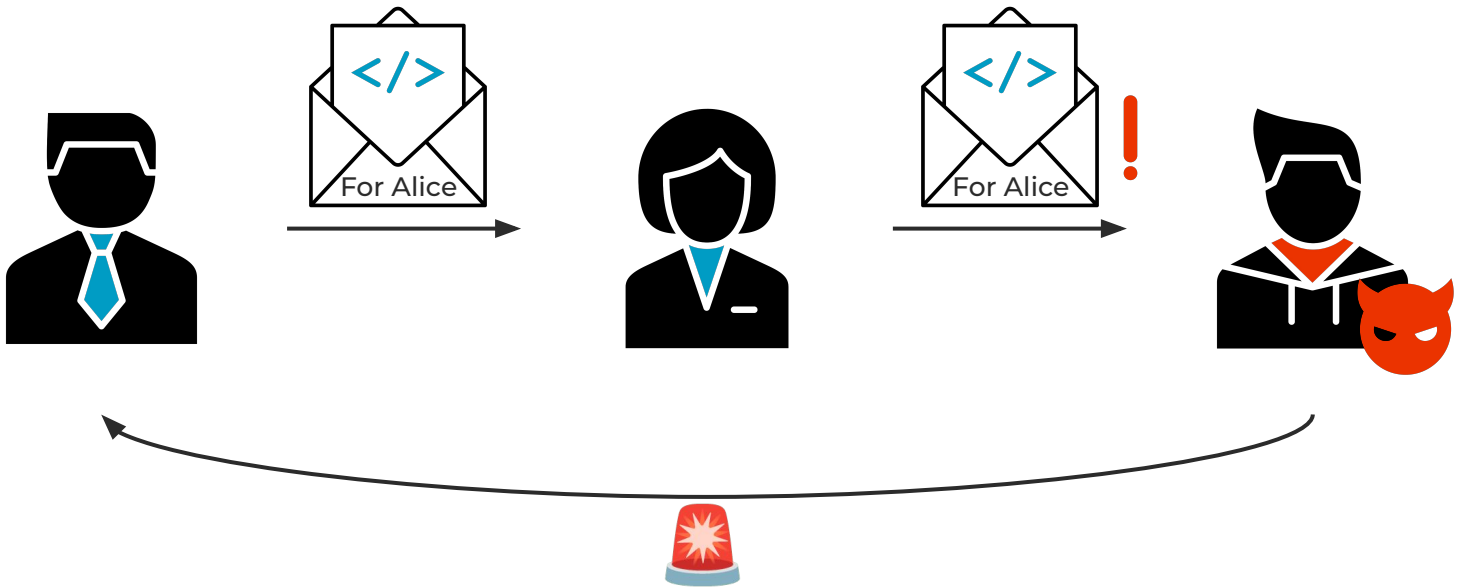
CISPA
HELMHOLTZ CENTER FOR INFORMATION SECURITY

Beauty at a Cost

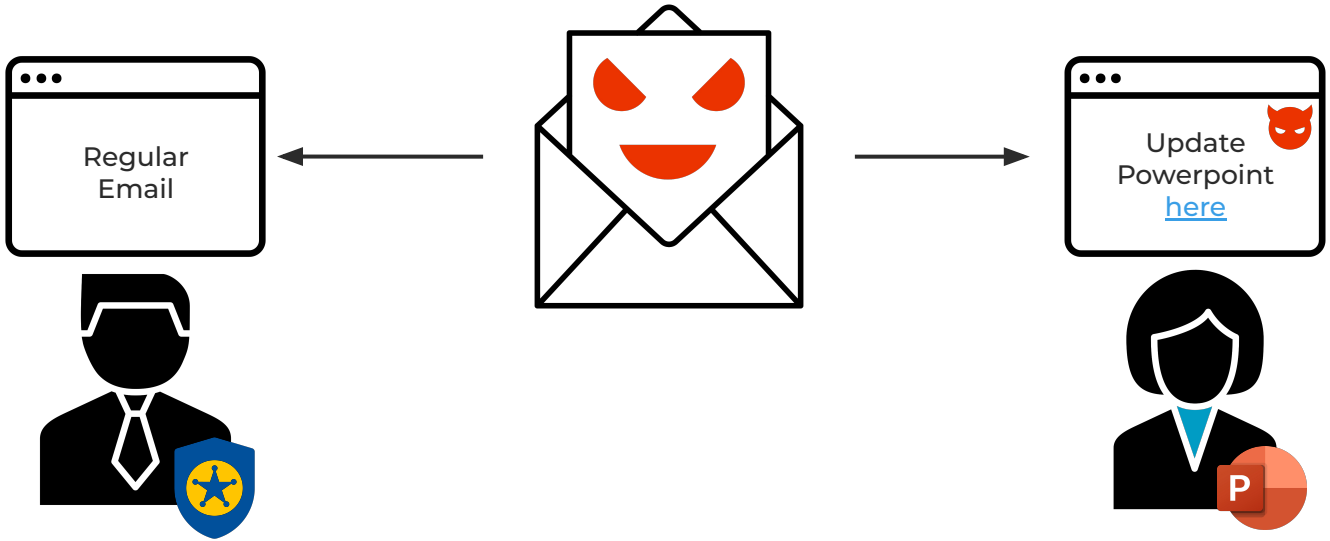
Privacy Implications of CSS on the Web and in Emails

Leon Trampert, **Daniel Weber**, Lukas Gerlach, Christian Rossow, Michael Schwarz

Email Forward Detection



Hidden Phishing Emails



Agenda

Browser Fingerprinting

CSS-based Browser Fingerprinting

Email Client Fingerprinting

$whoami - Leon Trampert

PhD Student

@ CISPA Helmholtz Center
for Information Security

Focus on

- Browser Security
- Side-Channel Attacks

Contact

leon.trampert.me

@ltrampert

$whoami - Daniel Weber

PhD Student

@ CISPA Helmholtz Center
for Information Security

Focus on

- CPU Security
- Side-Channel Attacks

Contact

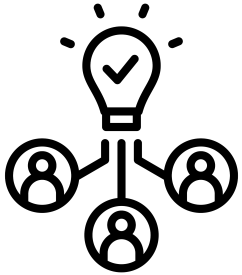
d-we.me

@weber_daniel

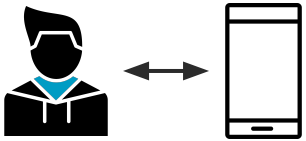
Browser Fingerprinting

What Is Browser Fingerprinting?

Identify Users

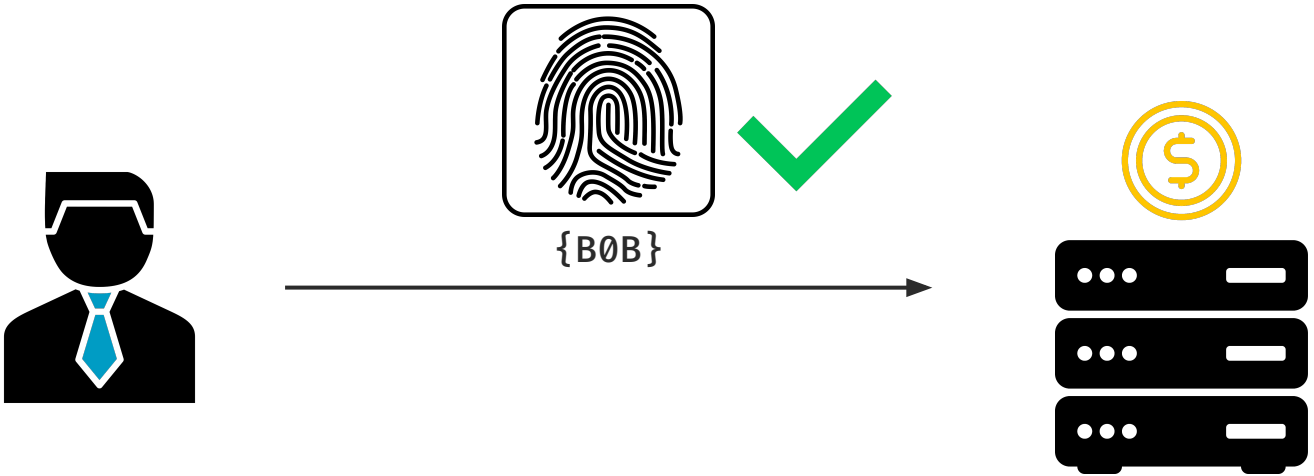


Link Sessions

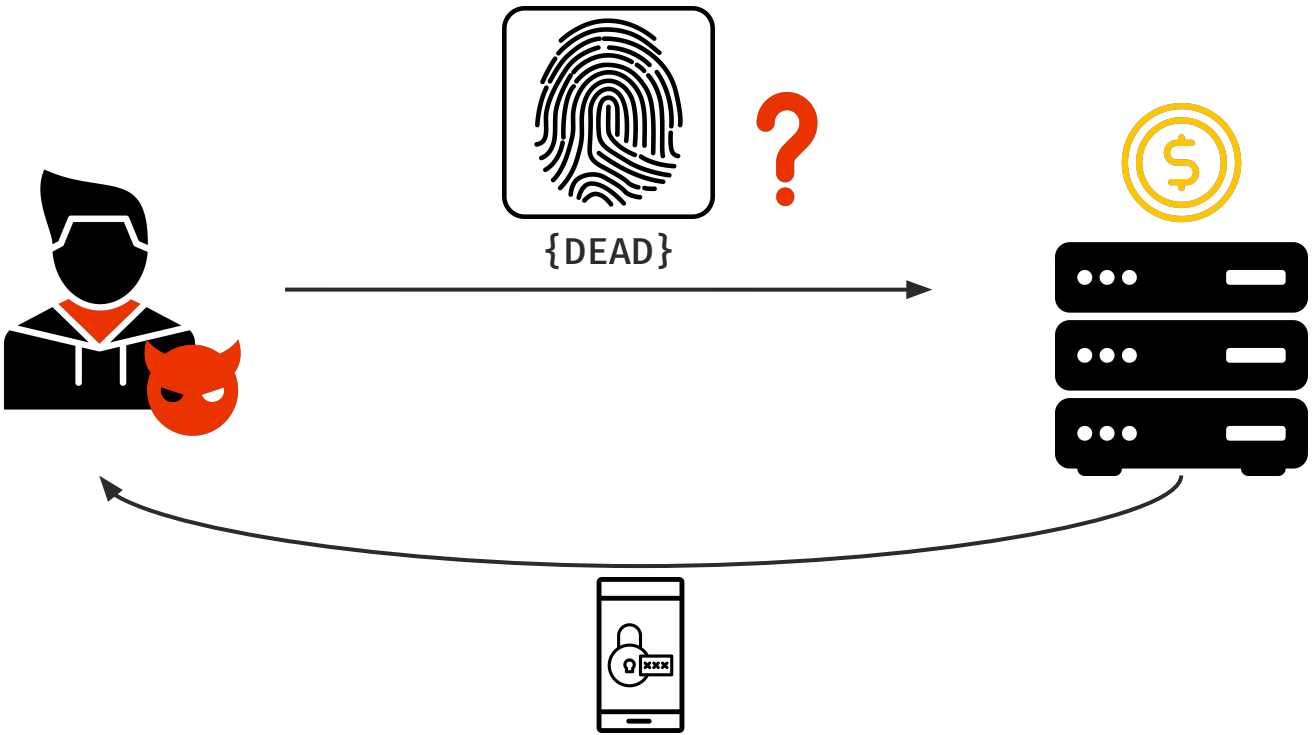


Link Users and Devices

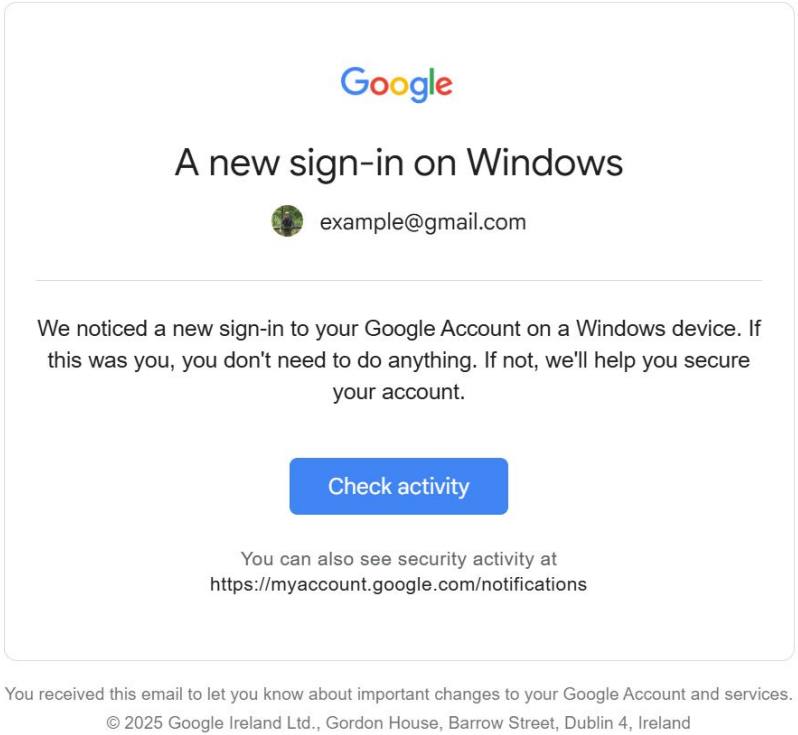
The Good – Risk-Based Authentication



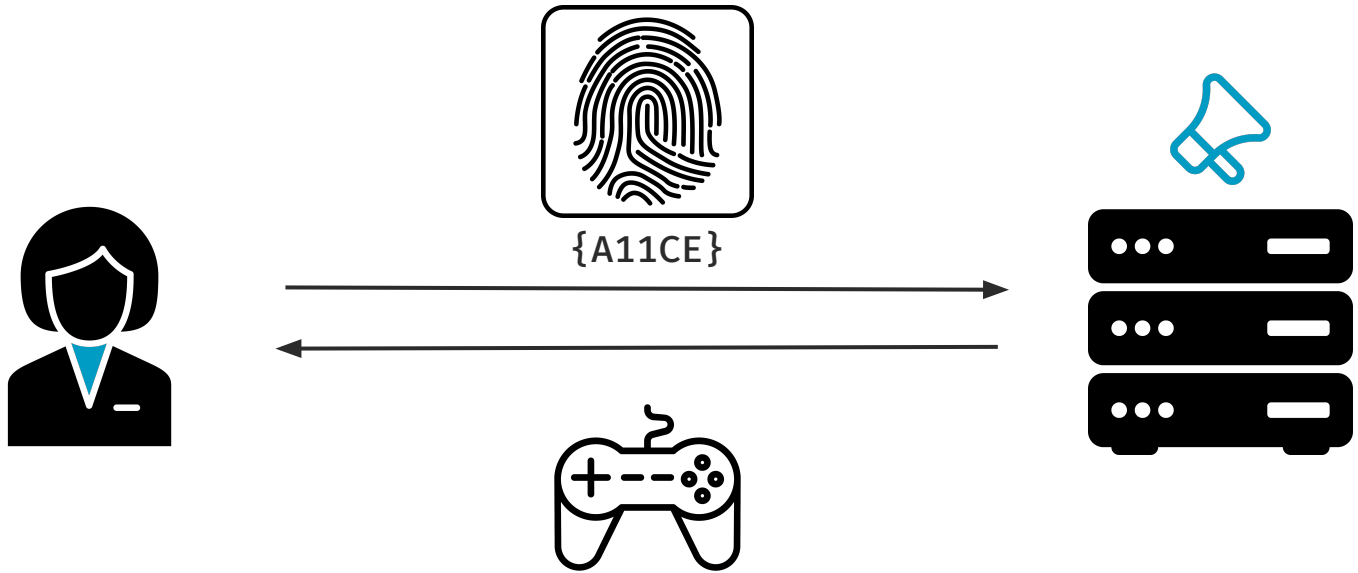
The Good – Risk-Based Authentication



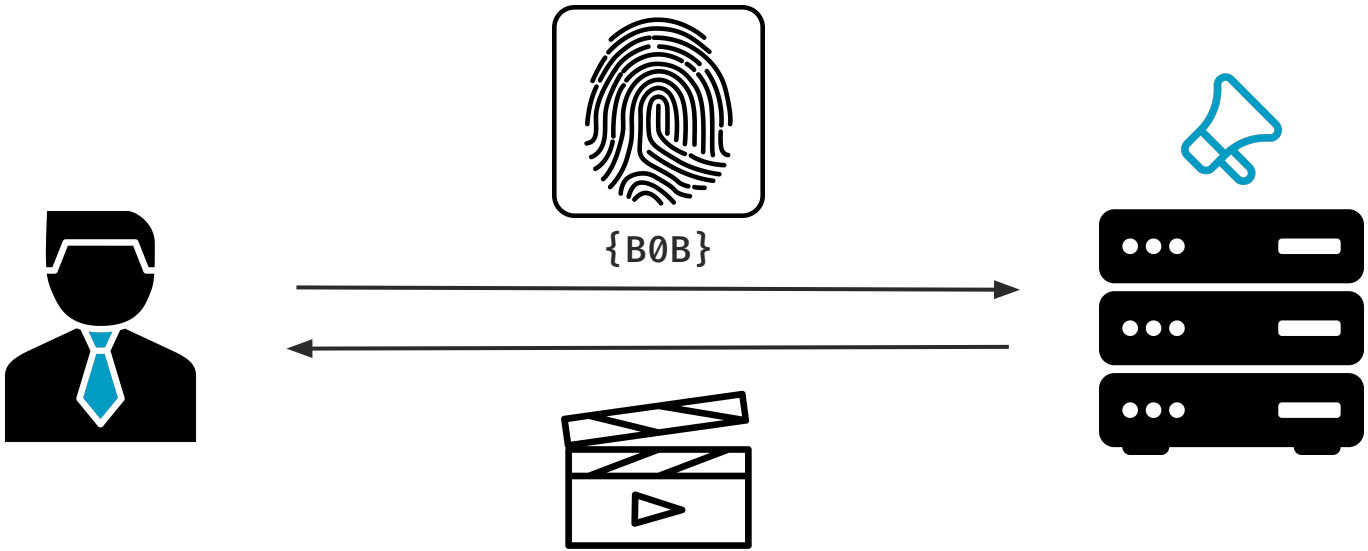
Example Notification



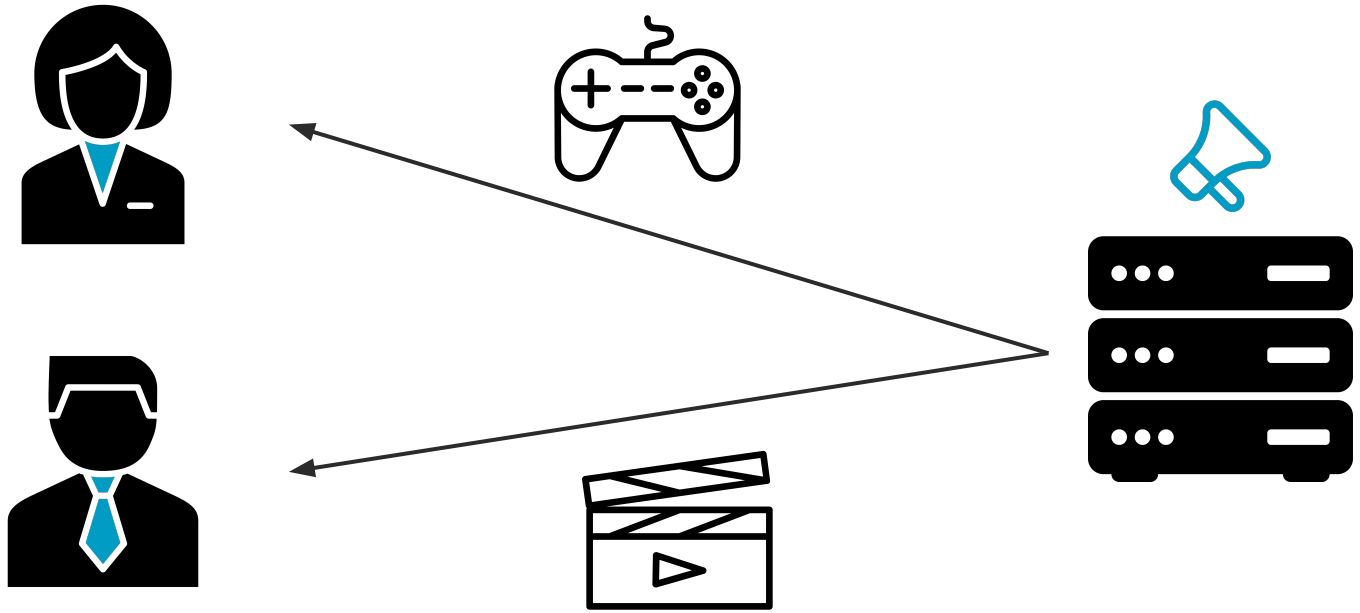
The Bad – User-Specific Web Content



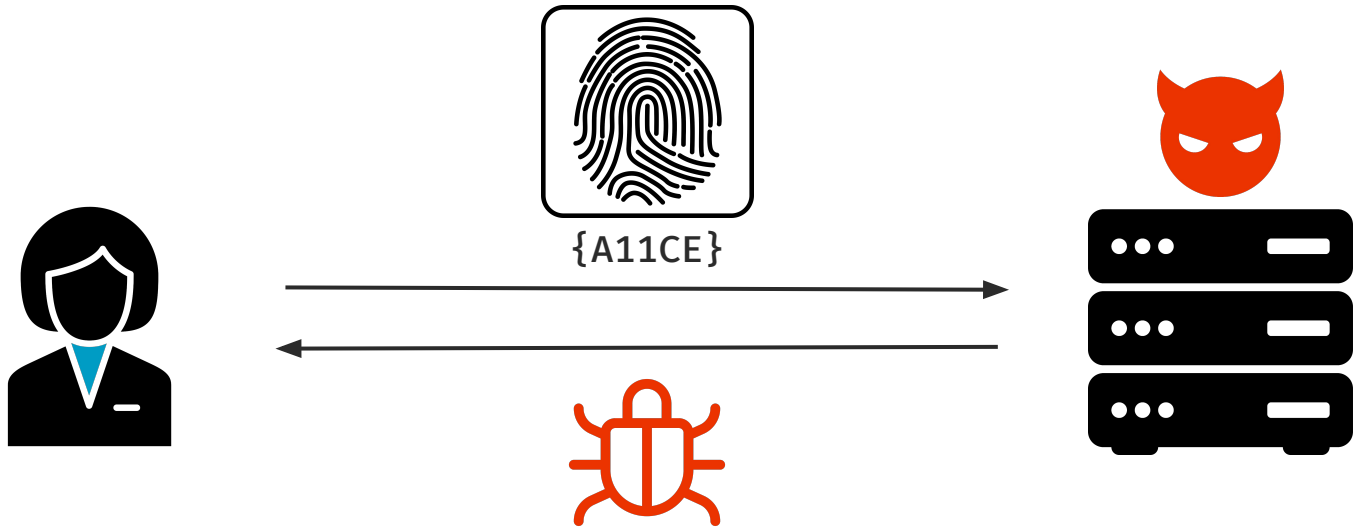
The Bad – User-Specific Web Content



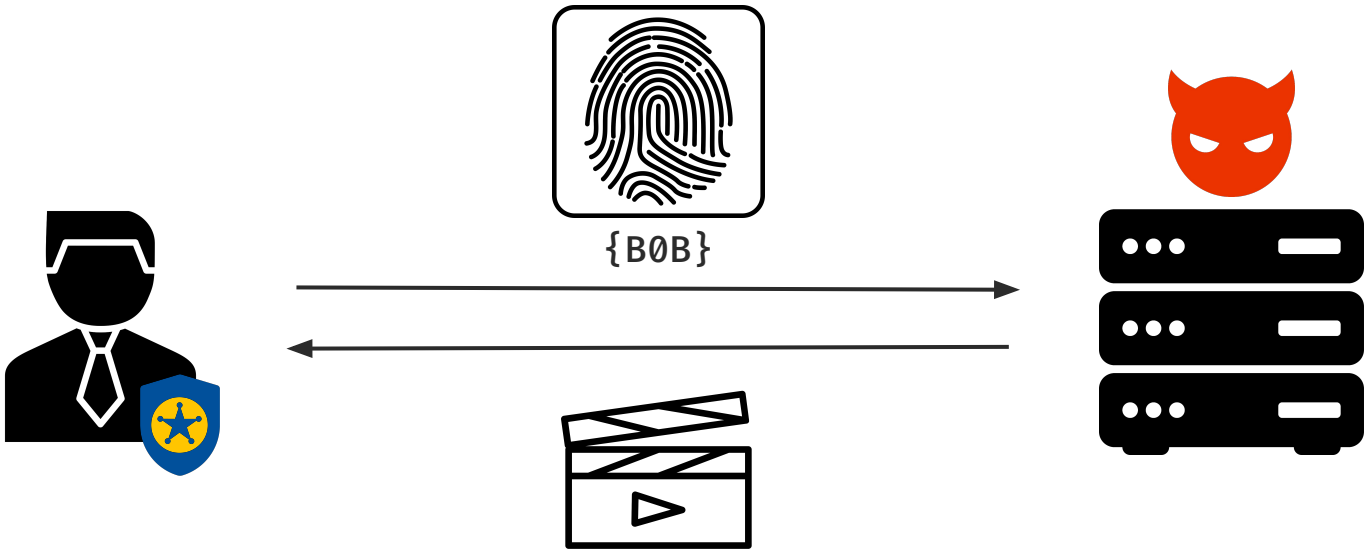
The Bad – User-Specific Web Content



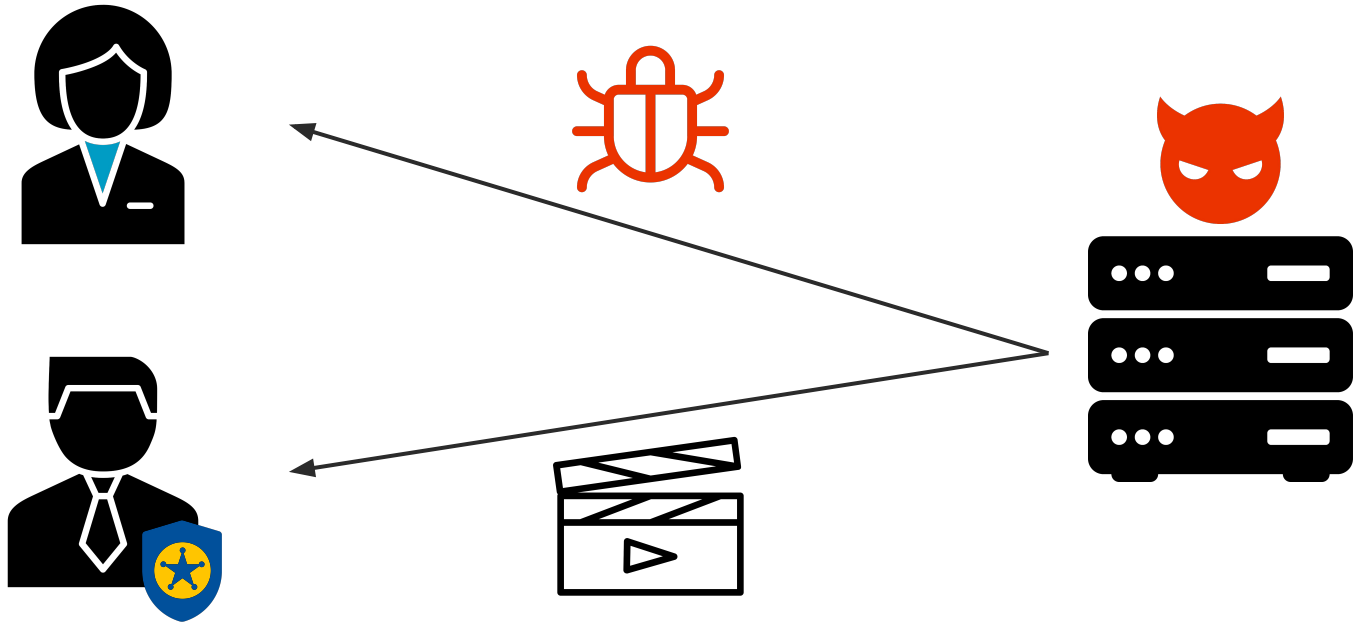
The Ugly – Hiding Malicious Content

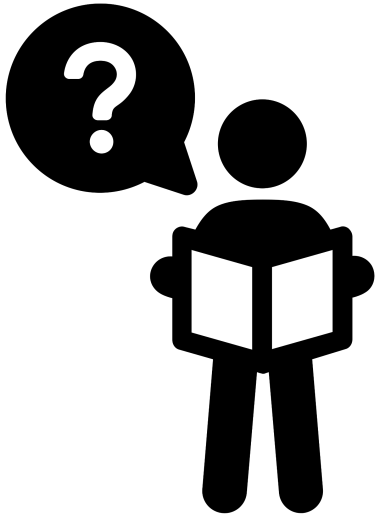


The Ugly – Hiding Malicious Content



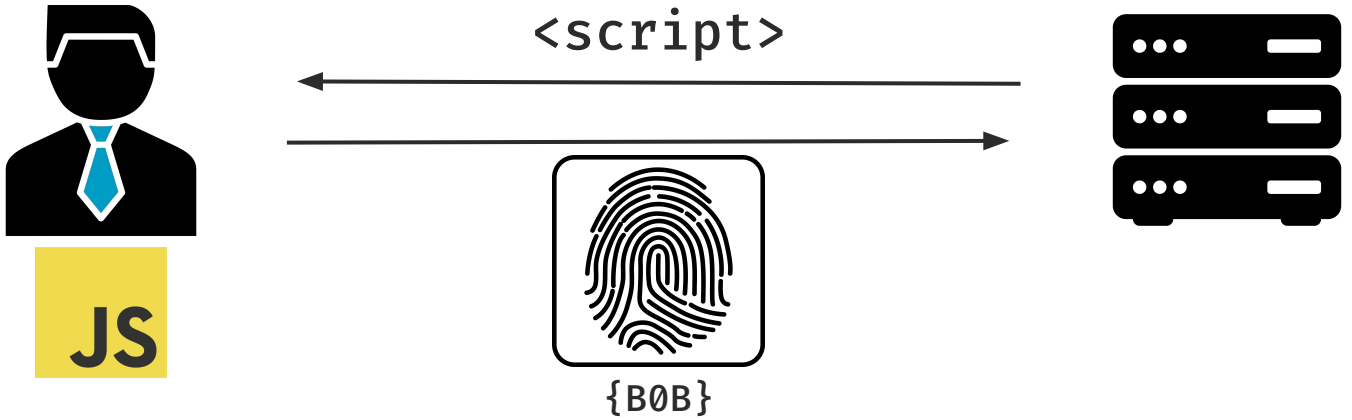
The Ugly – Hiding Malicious Content





How Does Fingerprinting Work?

Browser Fingerprinting



Font Fingerprinting

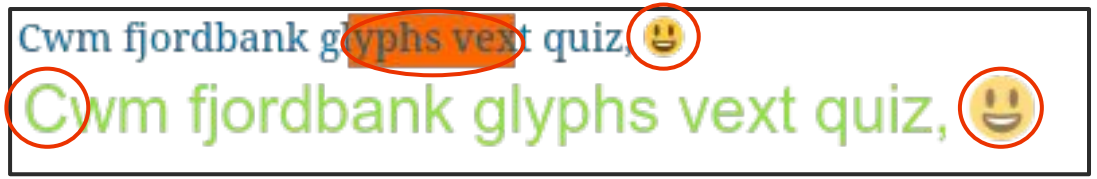
- Arial
- Comic Sans
- Pacifico

{A11CE}

- Arial
- Comic Sans
- **Gill Sans**
- Pacifico

{B0B}

Canvas Fingerprinting



{A11CE}

{B0B}

Current techniques rely on JavaScript!

Contexts without JavaScript

Tor Browser

NoScript

Email Clients

Can we fingerprint here?

Web Technologies

Web Technologies

CSS – Deep Dive

Cascading Style Sheets (CSS)

```
button {
 color: red;
 background-image:
   url(/pattern.png);
 width: calc(100% - 20px);
}
@media (min-width: 720px) {
 button {
   width: 100%;
 }
}
```

Cascading Style Sheets (CSS)

```
button {
 color: red;
 background-image:
   url(/pattern.png);
 width: calc(100% - 20px);
}
@media (min-width: 720px) {
 button {
   width: 100%;
 }
}
```

1. Properties

Cascading Style Sheets (CSS)

```
button {
 color: red;
 background-image:
   url(/pattern.png);
 width: calc(100% - 20px);
}
@media (min-width: 720px) {
 button {
   width: 100%;
 }
}
```

1. Properties
2. Functions

Cascading Style Sheets (CSS)

```
button {
 color: red;
 background-image:
   url(/pattern.png);
 width: calc(100% - 20px);
}
@media (min-width: 720px) {
 button {
   width: 100%;
 }
}
```

1. Properties
2. Functions
3. Selectors

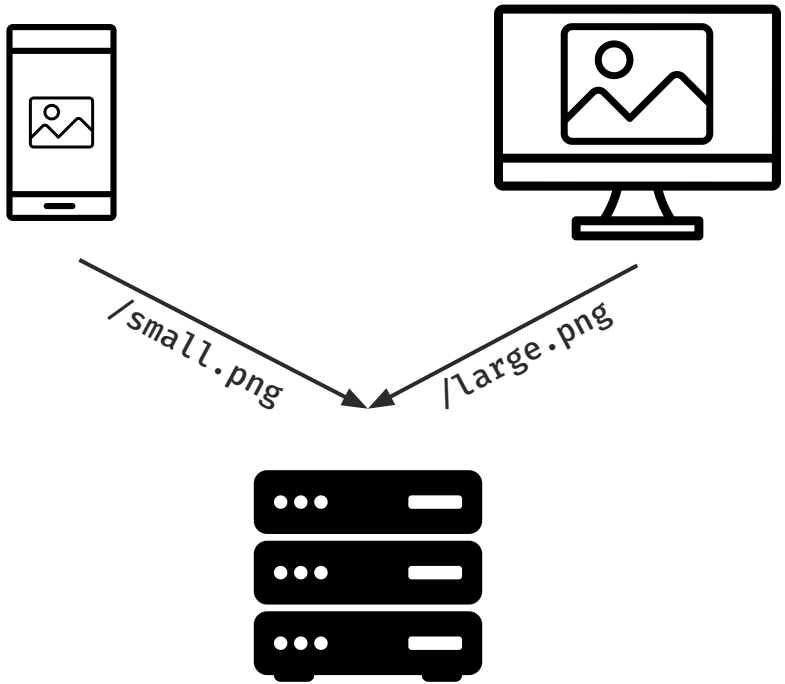
Cascading Style Sheets (CSS)

```
button {
 color: red;
 background-image:
   url(/pattern.png);
 width: calc(100% - 20px);
}
@media (min-width: 720px) {
 button {
   width: 100%;
 }
}
```

1. Properties
2. Functions
3. Selectors
4. @-rules

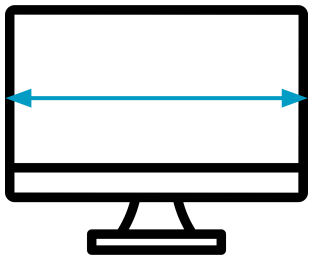
CSS-to-Server Communication

```
button {
 background-image:
          url(/small.png);
}
@media (min-width: 720px) {
 button {
   background-image:
           url(/large.png);
 }
}
```



Fingerprinting with CSS

Fingerprinting with CSS



@rules

@-Rules

@media

```
@media (min-width: 720px)
@media (prefers-color-scheme: dark)
@media (any-hover: hover)
```

Device Information & User Preferences

@supports

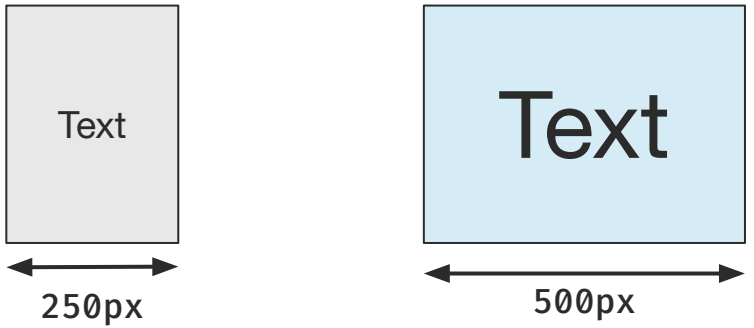
```
@supports (-moz-orient: block)
```

CSS Feature Support

New Rule: @container

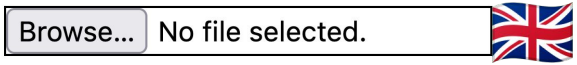
```
@container (width > 400px) {
  p {
    font-size: 16px;
    background-color: blue;
  }
}
```

`@container` is similar to `@media` but the queries are **relative to a container element**.

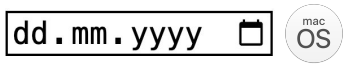


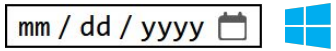
How is this useful?

Width Measurements!



Client Language

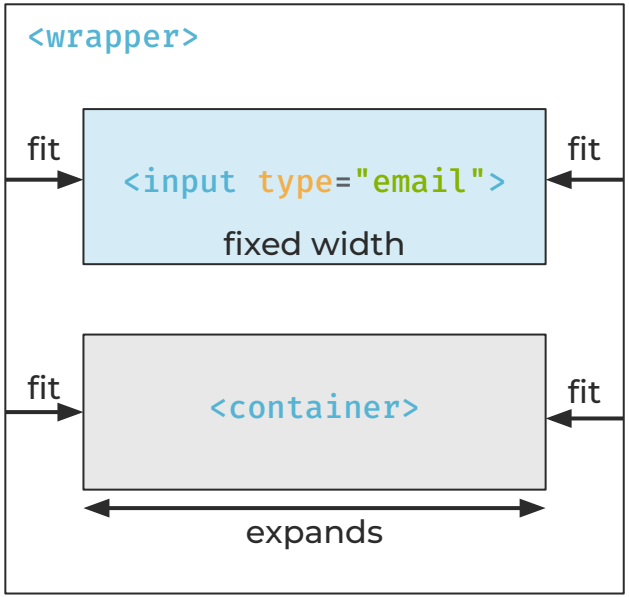




Browser / OS

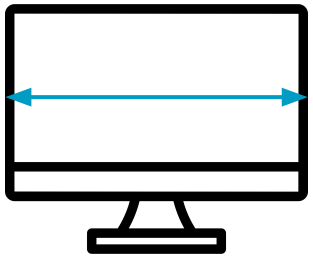
Fonts

@container – Example

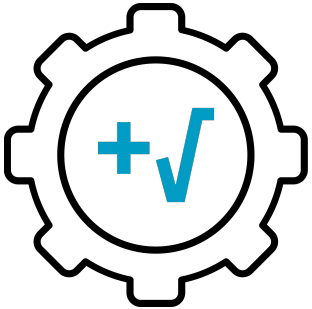



```
@container (width: 185px) {
  * {
      background-image:
          url(/ubuntu);
  }
}

@container (width: 177px) {
  * {
      background-image:
          url(/windows);
  }
}
```

Fingerprinting with CSS



@rules



Functions

Math Functions – Browser

```
calc(1px * (86566.45386119014 * sin(66505.33096836359 *
    251466.77293811357 - -8446.477528413574 / pi) *
     23954.456433470754 + 74259.77275980575 / pi))
```

0

1.78957e+7

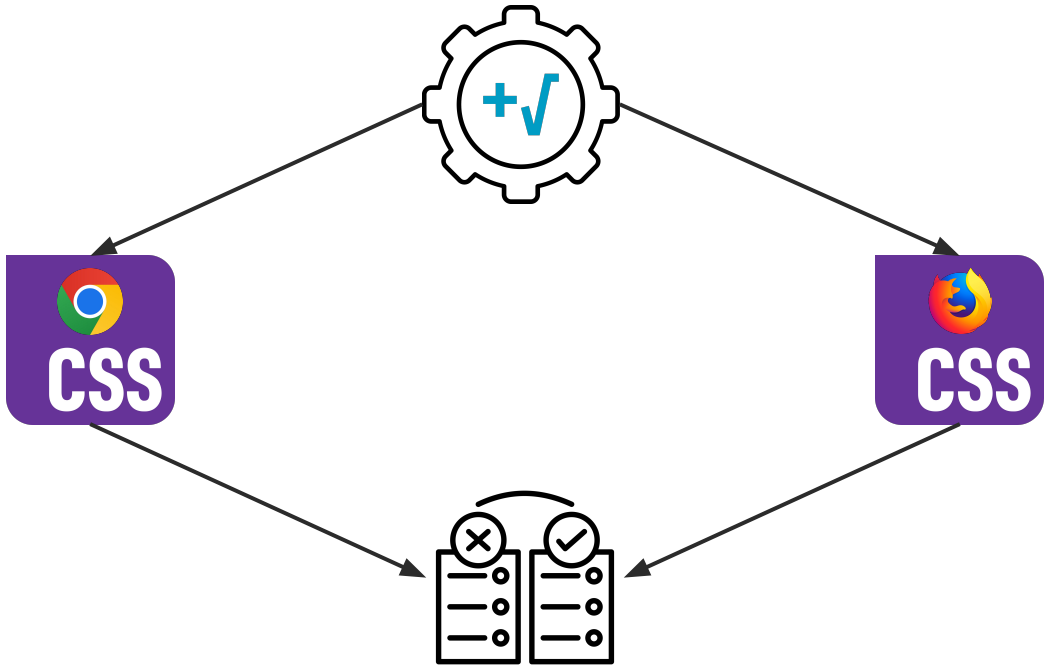
Math Functions – Architecture

```
calc(1px * (pi * pi + pi))
```

→ 13

→ 13.0104

Mental Arithmetic? Fuzzing!

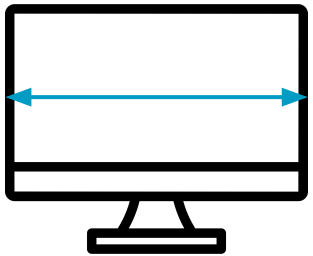


Math Functions – Example

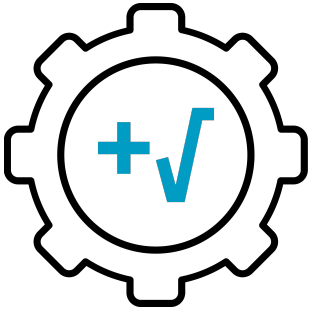
```
#target {
  width: calc(
    1px *
      (
        86566.45386119014 *
        sin(
          66505.33096836359 *
          251466.77293811357 -
           …
        )
      )
  );
}
```

```
@container (width: 0px) {
  * {
      background-image:
           url(/chrome);
  }
}

@container (width > 0px) {
  * {
      background-image:
           url(/firefox);
  }
}
```

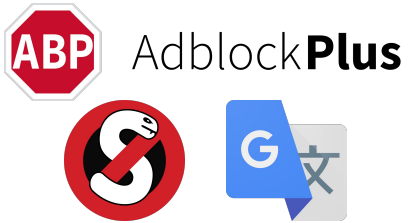
Fingerprinting with CSS



@rules



Functions



Plugin Detection

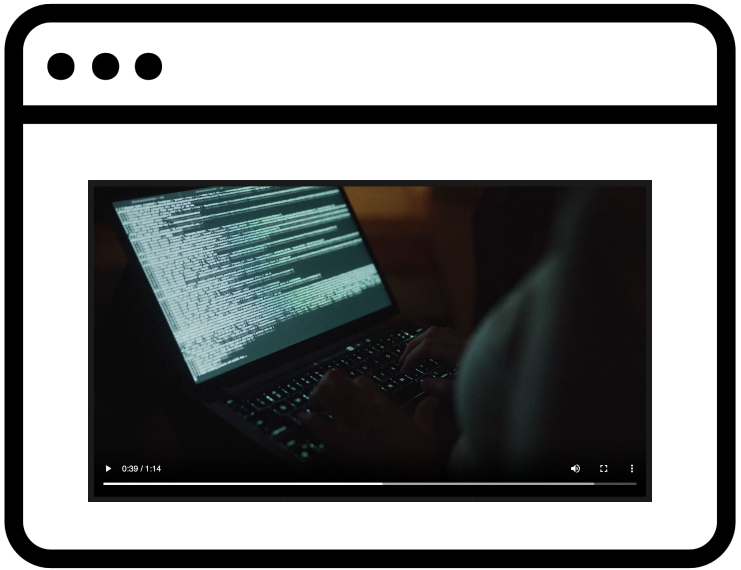
Plugin Detection

AdblockPlus

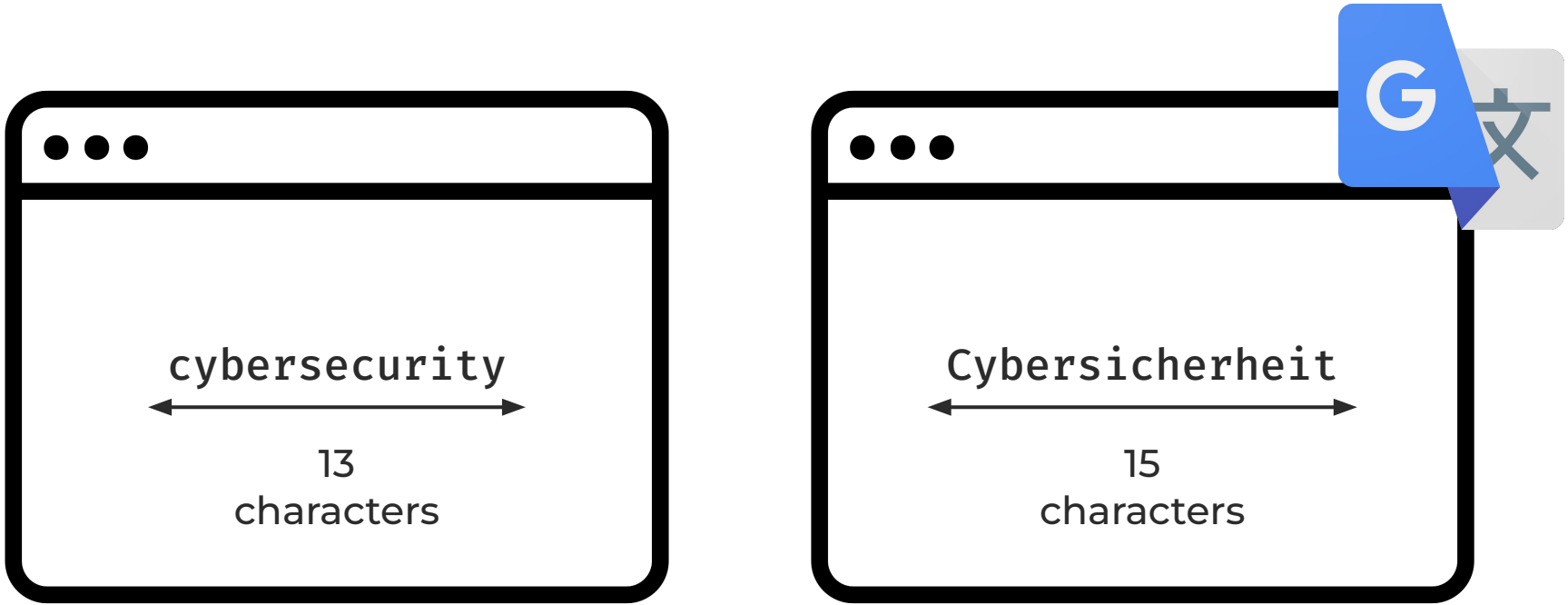
A user can **customize** their browser with Browser Extensions.

Browser Extensions and Translation Tools can **modify the content** of a website.

Plugin Detection – NoScript



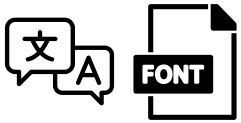
Plugin Detection – Google Translate



CSS Fingerprints

System Information

Hardware Information



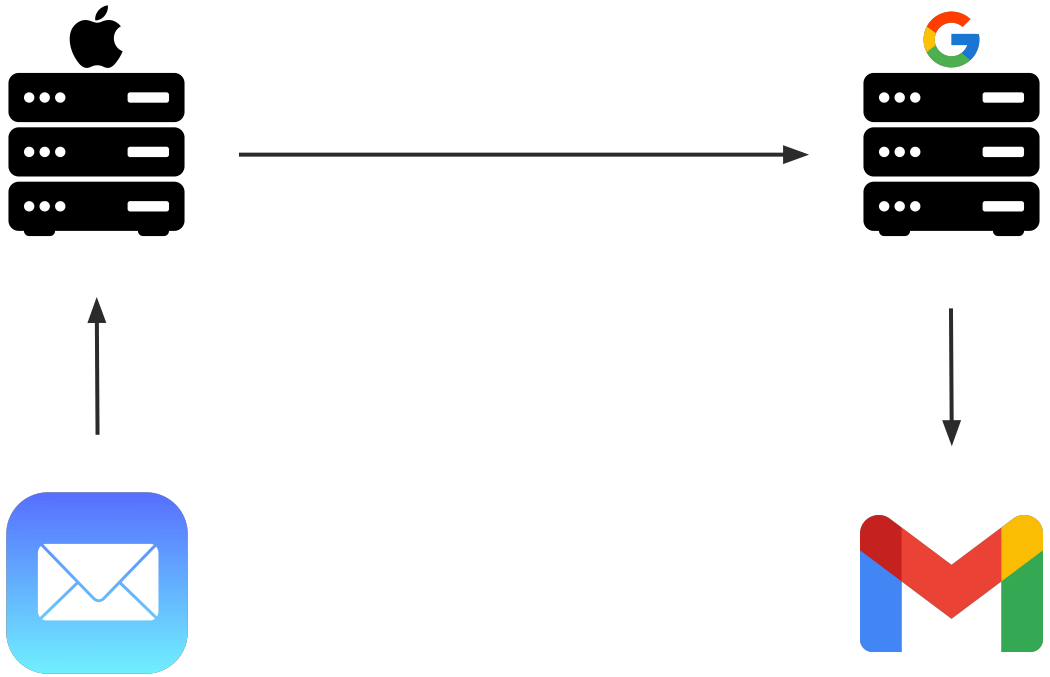
User Information

Plugin Detection

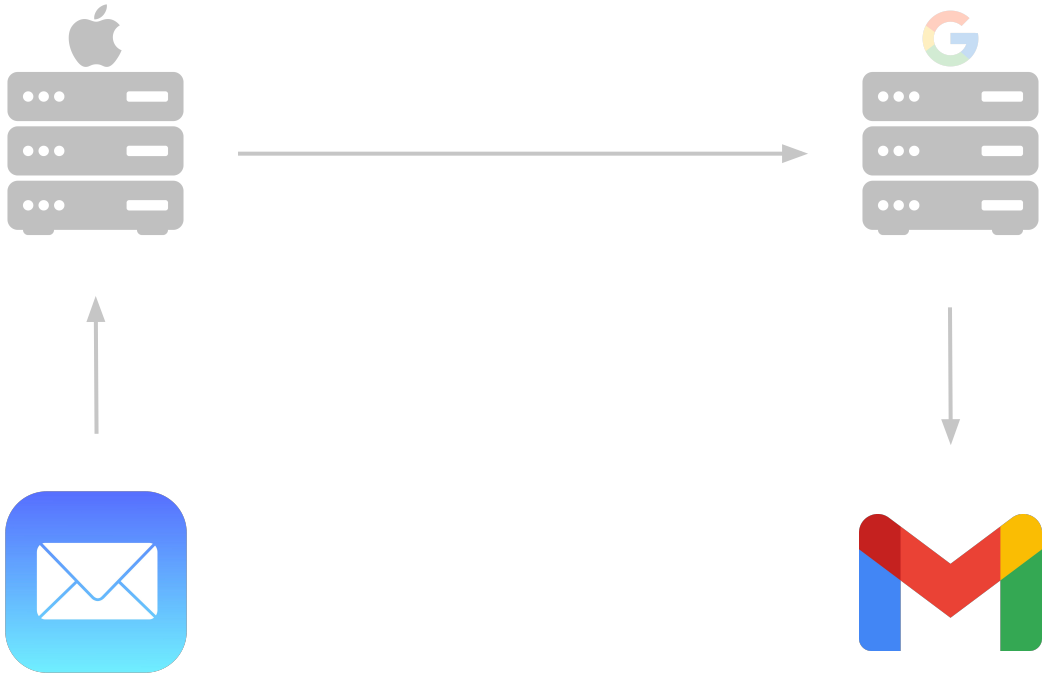
It works in the browser.

What About Email Clients?

Emails

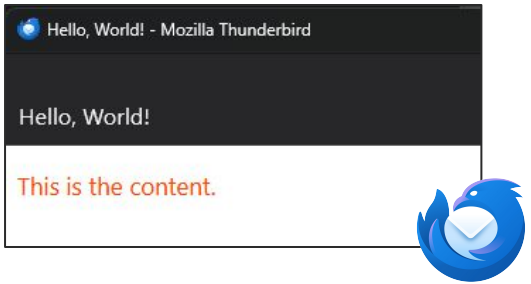


Emails

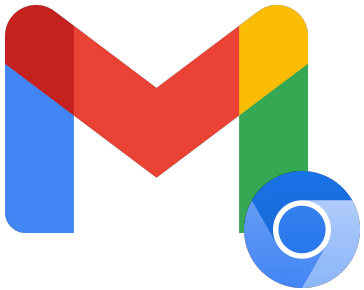


HTML Emails

```
Content-Type: text/html
Subject: Hello, World!

<html>

  <head>
    <style>
      p { color: orangered; }
    </style>
  </head>

  <body>
    <p>This is the content.</p>
  </body>

</html>
```



What Is an Email Client?



Webmail

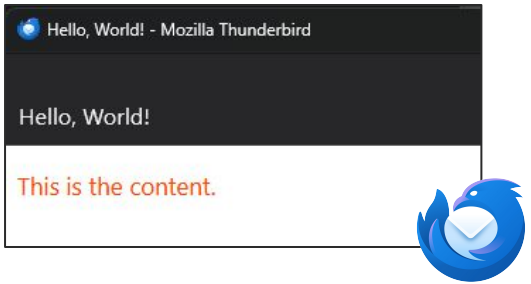
Desktop Clients

Mobile Clients

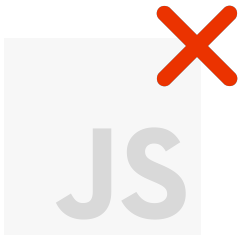
They are essentially browsers!

HTML Emails

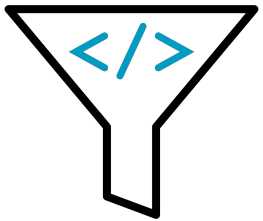
```
Content-Type: text/html
Subject: Hello, World!

<html>

  <head>
    <style>
      p { color: orangered; }
    </style>
  </head>

  <body>
    <p>This is the content.</p>
  </body>

</html>
```



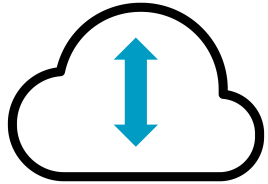
No Rules?



No JavaScript!



Subset of Features

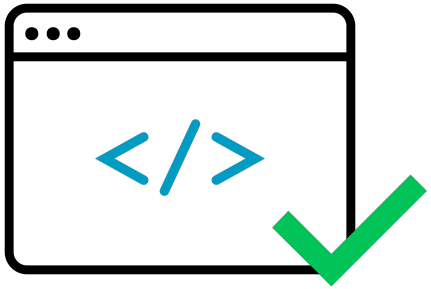


Proxy Servers

There is no standard for HTML emails!

So... Email Fingerprinting?

Fingerprinting in Emails?



CSS Feature Availability

Remote Content Loading

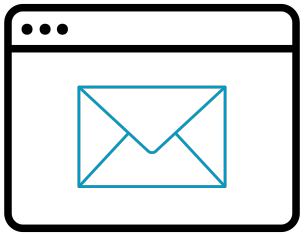
Analysis Time

Analysis of Clients

```
<body>
 <style>
   @supports (color: red) {
     p {
background-image: url(/supports-body);
     }
   }
 </style>
 <p>Text</p>
</body>
```

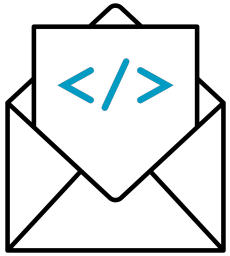
1. Conditions that are always true
2. Different **methods** of defining styles
   - `<style>` tags
   - `<link>` tags
   - `@import` directive
3. Different tag **locations**

Analysis of Clients



9 Webmail Clients
4 Desktop Clients
4 Android Clients
4 iOS Clients

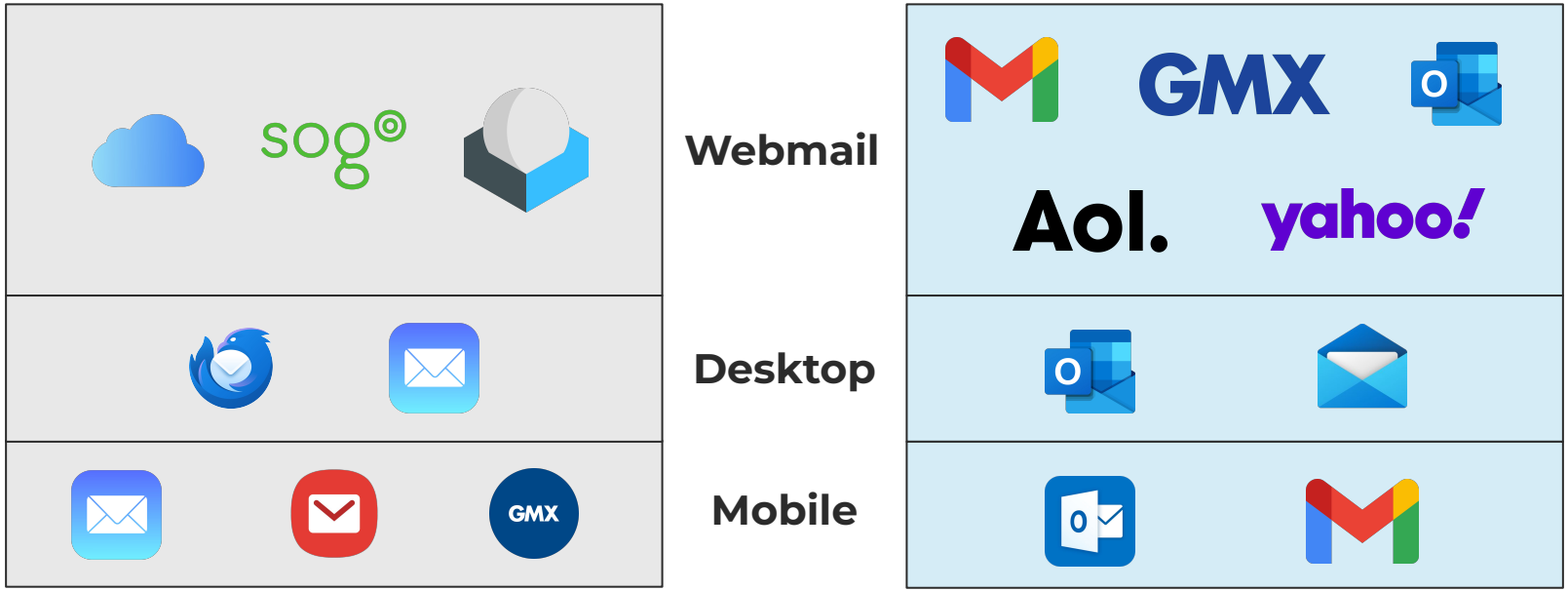
= 21 Clients in Total



100 Test Emails

= 2.100 Emails in Total

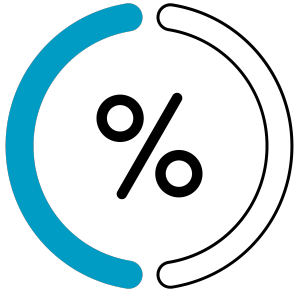
Client Behavior



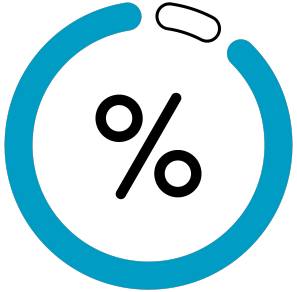
Lenient Clients

Restrictive Clients

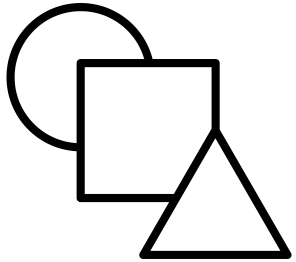
Email Client Fingerprinting



Half of clients allow **all** techniques



Almost all clients allow **some** techniques



Each client supports **different features**

Special Behaviors

Special Behavior – The Good

Behavior:

1. Allows almost **all CSS features**
2. Remote resources are loaded by a proxy **unconditionally**

Observation
Loading **all** resources prevents information leakage

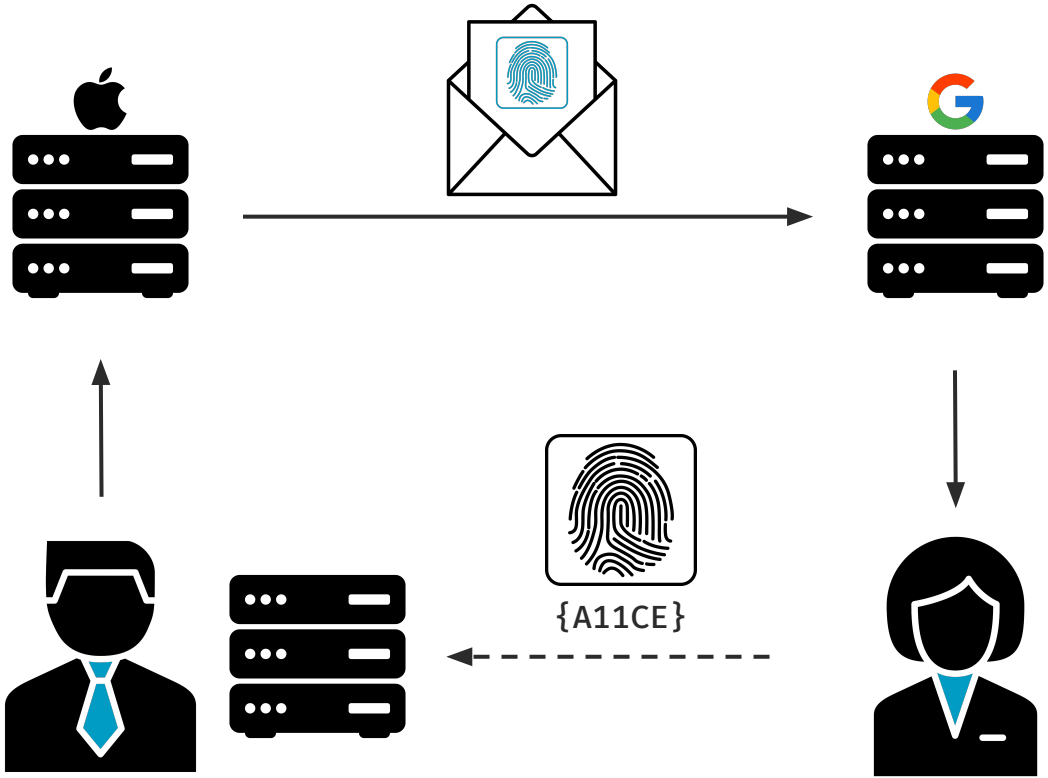
Special Behavior – The Bad

JavaScript in iFrames
(Bug Bounty)

CSS Injection
(CVE-2024-24510)

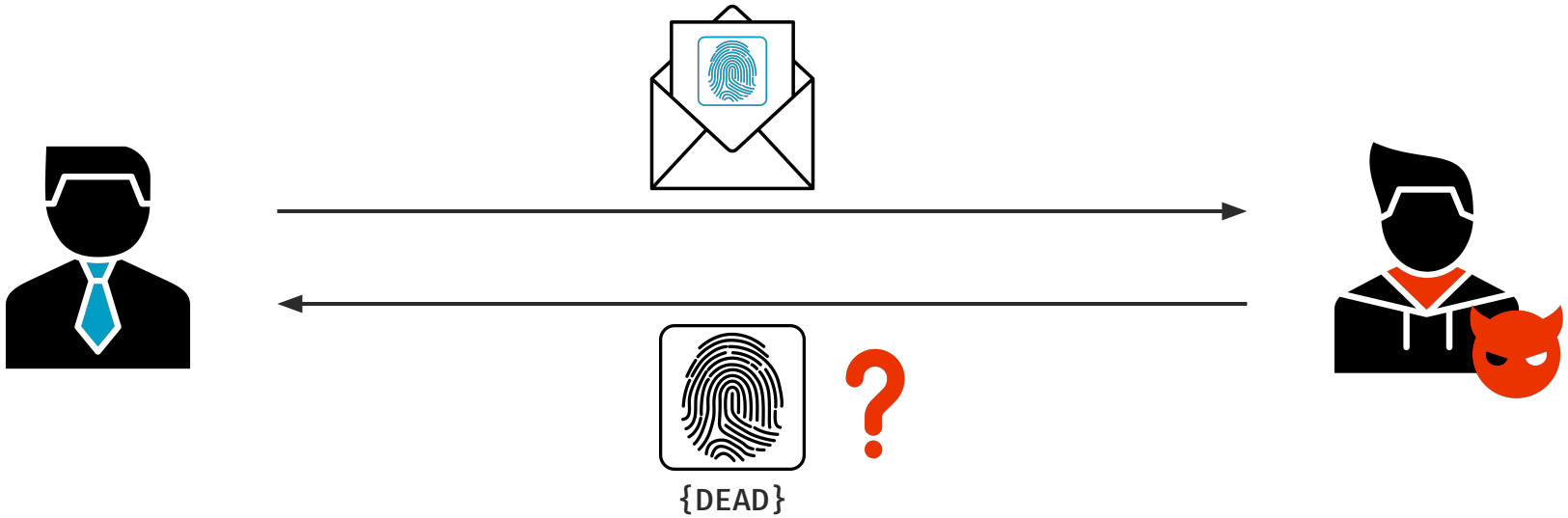
How Is Email Fingerprinting Useful?

Setup



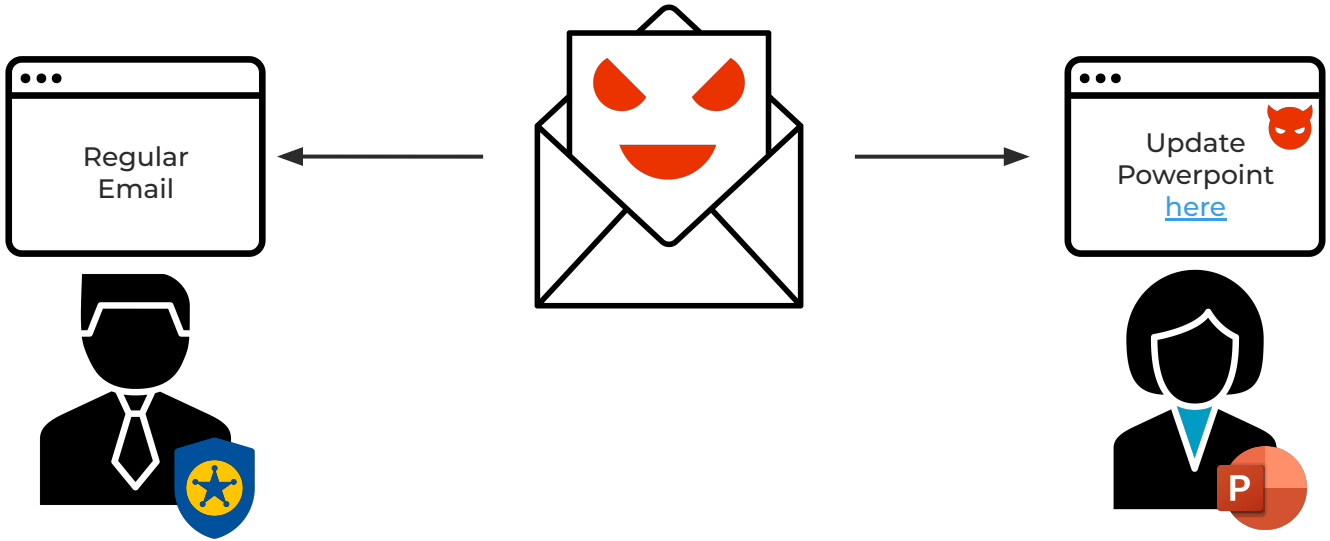
The Good – Leak Detection

The Good – Leak Detection



The Bad – Enhanced Tracking

The Ugly – Spear Phishing



Email Fingerprinting – Example

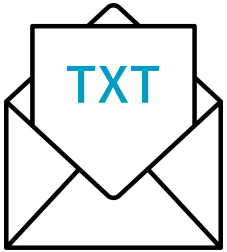
```
Content-Type: text/html
Subject: Office Detection

<html>
<body>

  <div id="target">
      <div id="yes"></div>
  </div>

  <style> … </style>

</body>
</html>
```

```
#target {
  container-type: inline-size;
  font-family: 'Gill Sans';
  width: 1cap;
}

@container (max-width: 7.5px) {
  #yes {
    background-image:
        url(/office-installed);
  }
}
```

Can We Prevent It?

Mitigations

Prevent Remote Content Loading



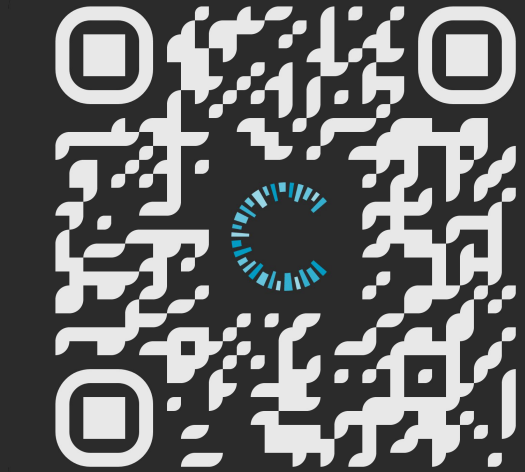
Plaintext Emails

Use a Restrictive Client

Wrapping Things Up

Takeaways

1. Browser Fingerprinting is a powerful tool for tracking, attacking and defending.
2. Stylesheets (CSS) alone can be used for Fingerprinting.
3. CSS-based Fingerprinting can even be used in emails.



s.roots.ec/spy-sheets