

WebBundles .wbns



CSSE340: Fundamentals of Cybersecurity
Team: siliconninja, Thomas Nandola, and Ricardo Hernandez

Hello Everyone. Thank you for joining us today. For our project, we decided to research further into Web Bundles. Web Bundles are a fairly new feature that Google has implemented. Today web bundles are only available in the Chrome browser, but potentially can be adopted by other developers in the future.

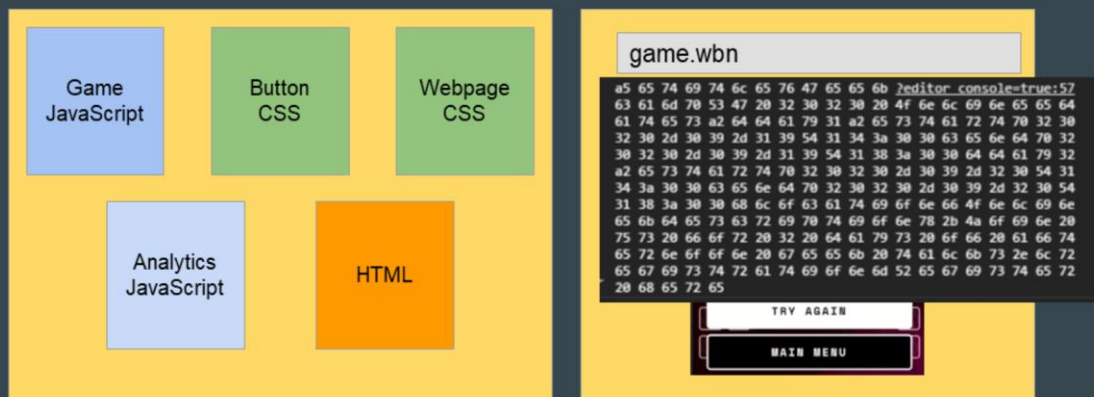
Overview

- I. Introduction
- II. What are Web Bundles?
- III. How do native and web applications differ from Web Bundles?
- IV. Motivation behind web bundles.
- V. CBOR Formatting
- VI. Threats (CIA)
- VII. Mitigation
- VIII. Usability of Mitigations

Ricardo:

Here is an overview of what we will be discussing.

Introduction | Web Bundles



Thomas: To begin, with Web Bundles we can package multiple web files (js, css, html) into the final webpage into a single file which you see on your right. It uses a specially formatted webpage using the CBOR format which compacts and jumbles code so it becomes extremely difficult for humans to read, (so you don't cheat in the game) which we'll go into detail about later. However, there have been some concerns raised about the possible exploits of this new technology that we will also go into further in this paper.

No demonstrations of CBOR, and packing into single file.

What are Web Bundles?

- A Concise Binary Object Representation (CBOR) file with a “.wbn” extension, which is faster to send and decode.
- Bundled HTTP Exchanges: File that contains set of HTTP requests
- Allows for a self-contained web app that can be shared offline or over low-bandwidth connections
- Contains:
 - HTML files
 - CSS stylesheets
 - Javascript Files
 - Images
 - Etc.

siliconninja:

... As you saw on the previous slide, the bundle contains all these web files. It's an improvement over separate HTTP requests because it's faster...

How do Native and Web Applications differ from Web Bundles?

Native Applications

- Run on particular operating system
- Stand-alone
- Don't use much bandwidth
- Large files at times

Web Applications

- Require internet connection
- Runs on web server
- Potentially slower load times over internet

Web Bundles

- Package up website resources (HTML, CSS, Javascript, Images, Fonts)
- Bundles a group of HTTP exchanges
- Don't need internet connection
- Can be transferred easily by any means (USB, NFC, Bluetooth, etc)
- Save data costs and make loading times instantaneous when served locally
- Currently requires Chrome 80 or later for loading, still experimental

Thomas:

EXAMPLES:

Native Application: Microsoft Word

Web App: Microsoft Word in your browser

Web Bundle: you saw an example earlier; it combines the best of both worlds in some cases

Important Definitions to Explain:

HTTP exchanges mean sets of requests and expected responses.

What's the motivation behind Web Bundles?

Ad Companies:

Harder to block ads \Rightarrow more money!

Google says:

Makes Web Applications like Native Applications

Lets you run things offline



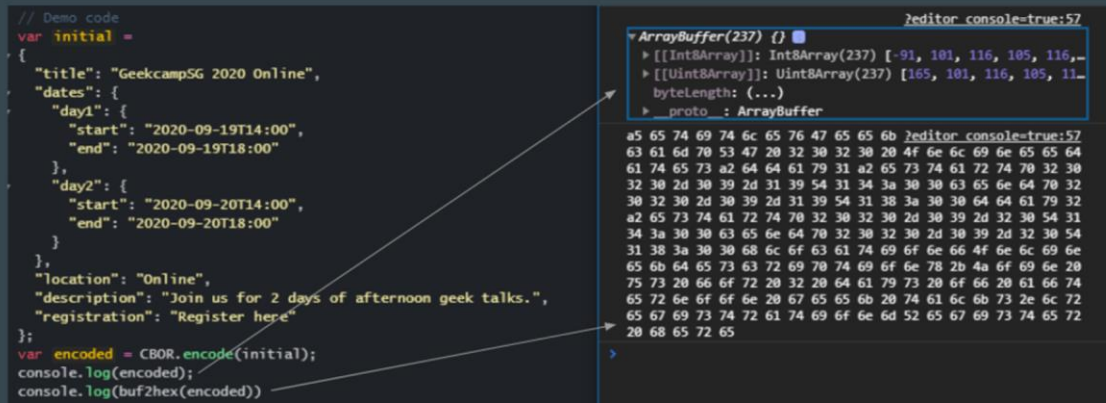
Ricardo

Randomization of URL's of the sources included in the bundle. For ad blockers, say we want to block a JS resource named ads.js because we know that we do not want that source loaded when we are executing the web application/bundle. However, because web bundles are, in a sense like PDF, that they are an all-or-nothing file. You either don't get any of the resources in the file or you get everything that is included in the bundle. Without a proper decoder to be able to inspect a web bundle, you aren't sure what is included in the web bundle. You can easily block */ads.js in your ad-blocker for a normal web application to stop the javascript from even downloading. This practice may prove difficult in the web bundle since you download the whole bundle, and the resource can have an arbitrary name (say "forcedads.js"). You could also reuse the URL and make it point to different resources in each bundle. Or you change the URLs to ones you know won't be blocked. Therefore, this allows users no control over what is being downloaded and executed when using a web bundle, so ad companies are able to bring in more ad revenue.

For example, if you can build bundles on-the-fly per request, then you really can make almost any content unblockable. While Google's tool does not have a bundle edit implemented, it could be done on an edge server, which are powerful computers put at the "edge" of a given network where data computation needs to happen. They are the closest to the systems or applications that are creating the data being stored on,

or used by, the edge server.

CBOR Formatting



```
// Demo code
var initial =
{
  "title": "GeekcampSG 2020 Online",
  "dates": {
    "day1": {
      "start": "2020-09-19T14:00",
      "end": "2020-09-19T18:00"
    },
    "day2": {
      "start": "2020-09-20T14:00",
      "end": "2020-09-20T18:00"
    }
  },
  "location": "Online",
  "description": "Join us for 2 days of afternoon geek talks.",
  "registration": "Register here"
};
var encoded = CBOR.encode(initial);
console.log(encoded);
console.log(buf2hex(encoded))
```

```
ArrayBuffer(237) {}
  ▶ [[Int8Array]]: Int8Array(237) [-91, 101, 116, 105, 116, ...]
  ▶ [[UInt8Array]]: UInt8Array(237) [165, 101, 116, 105, 116, ...]
  ▶ byteLength: (...)
  ▶ proto: ArrayBuffer

a5 65 74 69 74 6c 65 76 47 65 65 6b 2editor console=true:57
63 61 6d 70 53 47 20 32 30 32 30 20 4f 6e 6c 69 6e 65 65 64
61 74 65 73 a2 64 64 61 79 31 a2 65 73 74 61 72 74 70 32 30
32 30 2d 30 39 2d 31 39 54 31 3a 3a 30 30 63 65 6e 64 70 32
30 32 30 2d 30 39 2d 31 39 54 31 38 3a 30 30 64 64 61 79 32
a2 65 73 74 61 72 74 70 32 30 32 30 2d 30 39 2d 32 30 54 31
34 3a 30 30 63 65 6e 64 70 32 30 32 30 2d 30 39 2d 32 30 54
31 38 3a 30 30 68 6c 6f 63 61 74 69 6f 6e 66 4f 6e 6c 69 6e
65 6b 64 65 73 63 72 69 70 74 69 6f 6e 78 2b 4a 6f 69 6e 20
75 73 20 66 6f 72 20 32 20 64 61 79 73 20 6f 66 20 61 66 74
65 72 6e 6f 6f 6e 20 67 65 65 6b 20 74 61 6c 6b 73 2e 6c 72
65 67 69 73 74 72 61 74 69 6f 6e 6d 52 65 67 69 73 74 65 72
20 68 65 72 65
```

Ricardo

CBOR is based on the JSON data model which includes: numbers, strings, arrays, maps (called objects in JSON), and a few values such as false, true, and null.

This is an example encoding of an object with the title "GeekcampSG 2020 Online". It is first being encoded and then translated into Hex.

CBOR encoder must serialize data over network.

This data is encoded (usually in base64 format), adding complexity and bulk to the file.

CBOR is an archive format with a parsable index and easy-to-read individual files based on their offset in the bundle. This allows for significant faster processing and faster transferring speeds, but lacks human readability, which is what makes it difficult for the users (like us) to know what is included.

Do they cause a threat?

- Existing security related browser extensions won't work with Web Bundles
- Additional attack surface for CBOR decoders
- Can obfuscate things like JavaScript
- Easy to use so might increase adoption of this tool

Major Types	Unsigned integer + text string	Array + text string	Array + Map (JSON object)
Representation	Index 1: 97100115 Index 1000: '.js'	Index 1: ['a', 'd', 's'] Index 1000: '.js'	Index 1: ['a', 'd', 's'] Index 1000: {'i': 115}

Thomas:

Second bullet: it is a component on top of the web application

This is a way that CBOR can jumble and randomize URLs which is called obfuscation.

Ad.js, for example, malicious actors can utilize this tool to make their code unreadable so others cannot figure out what they may be doing.

Also a lot of the spec does trust that CBOR encoders and decoders will do the right thing.

On bottom: there are many ways in the parsable index of the CBOR file that a simple location to a JavaScript file (ads.js) can be loaded and executed.

For example, you can use different data types to encode different parts of the URL, making it hard for existing security tools, such as ad-blockers, to simply find "ads.js".

These variations not only decrease file size in some cases, but also can obfuscate the URL or the set of HTTP requests.

Additionally, the tool itself is easy to use which can increase adoption and thus the threats associated with it.

Threats - Confidentiality

- Confidentiality of data on the OS it's running on is compromised
 - Can execute more malicious instructions faster with CBOR format vs. JS
- Existing security tools can't decrypt URLs inside a Web Bundle
 - Easier to perform something like MITM attacks on communications

Threats - Integrity

- CBOR decoders: can't prove they are executed in the browser's sandbox
 - TRUST the browser
 - Doesn't follow Simplicity
- Google (in Chrome) might not ensure the authenticity of HTTP exchanges by requiring signatures
 - WICG: This is being done
 - Will Google follow the standard?

Ricardo:

1st bullet point:

A sandbox is a tightly controlled environment where programs can be run. For example, Google Chrome and Internet Explorer both run in a sandbox themselves. These browsers are programs running on your computer, but they don't have access to your entire computer. They run in a low-permission mode. When decoding (CBOR) and executing the code, we are trusting that the browser is running the decoded CBOR byte stream in the browser's sandbox and not elsewhere in the user systems. There is no clear documentation on what standard or procedure that Google plans to implement with the decoding and execution of web bundles.

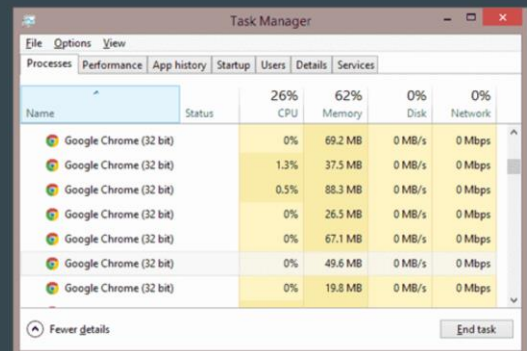
2nd bullet point:

According to WICG, the HTTP exchanges will be changed: there will be signatures on them to ensure integrity in transit (when sending data back/forth to servers).

HTTP resources in a Web Bundle are indexed by request URLs, and can OPTIONALLY come with signatures that vouch for the sources. Signatures allow browsers to understand and verify where each resource came from, and treats each as coming from its true origin. This is similar to how Signed HTTP Exchanges, a feature for signing a single HTTP resource, are handled. But Google might not follow this standard in Chrome. This is another important point of contention for integrity because many people use Chrome.

Threats - Availability

- Existing browser security tools won't work at all
- Untrusted origins (that execute from unsigned Web Bundles) can fill up browser memory more easily than JavaScript
 - Causes browser crashes



The screenshot shows the Windows Task Manager Performance tab. The 'Memory' column shows that several Google Chrome (32 bit) processes are using a significant amount of memory, with one process using 69.2 MB. The 'CPU' column shows that the same processes are also using a significant amount of CPU, with one process using 26%. The 'Disk' and 'Network' columns show that the processes are also using a significant amount of disk and network resources.

Name	Status	26% CPU	62% Memory	0% Disk	0% Network
Google Chrome (32 bit)		0%	69.2 MB	0 MB/s	0 Mbps
Google Chrome (32 bit)		1.3%	37.5 MB	0 MB/s	0 Mbps
Google Chrome (32 bit)		0.5%	88.3 MB	0 MB/s	0 Mbps
Google Chrome (32 bit)		0%	26.5 MB	0 MB/s	0 Mbps
Google Chrome (32 bit)		0%	67.1 MB	0 MB/s	0 Mbps
Google Chrome (32 bit)		0%	49.6 MB	0 MB/s	0 Mbps
Google Chrome (32 bit)		0%	19.8 MB	0 MB/s	0 Mbps

Thomas:

1st bullet point-

Lack of availability of these tools.

2nd bullet point -

Because it executes in binary format.

Continue sending http exchanges (request/responses) to overload cache - and slow.

Image - <https://ssl-proxy->

<https://ssl-proxy-updated.herokuapp.com/6ac407c723c05bb1d399d2faa51b6f1e8dbfd12e/687474703a2f2f7777772e746563686e6970616765732e636f6d2f77702d636f6e74656e742f75706c6f6164732f323031342f30342f476f6f676c652d4368726f6d652d70726f63657373657732d696e2d57696e646f77732d5461736b2d4d616e616765722e706e67/>

Mitigations - Fail-Safe Stance

- Sandboxes stop code from reading files
 - Web Browser
- Whitelists (allow lists) stop URLs from sending anything that could be malicious
 - OS, Network
- Antiviruses can detect suspicious requests
 - OS, Network

siliconninja:

Whitelists provide user control by allowing the user to choose webpages they deem non-malicious.

Can also be applied at the network level, but have to tell admin that you want to unblock new URLs, say if you're developing a new app and need URLs unblocked. Antiviruses can use heuristics to detect suspicious requests (detect stage).

Usability of Mitigations

- Whitelisting: could use existing sets of whitelists to improve usability
- Sandboxing generally usable
- Antiviruses: usable, but not perfect and expensive (!!)

siliconninja:

“Filter lists” provide most usability for this strategy. Does come with cost of trusting user, but easier than placing load on network admins.

Questions?

Image Sources

Slide 3: <https://www.youtube.com/watch?v=QGdBTtEWEmc>, <https://web.dev>

Slide 5: https://emojipedia-us.s3.dualstack.us-west-1.amazonaws.com/thumbs/240/twitter/259/thinking-face_1f914.png

Slide 7: <https://www.youtube.com/watch?v=QGdBTtEWEmc>

Slide 11: <https://ssl-proxy-updated.herokuapp.com/6ac407c723c05bb1d399d2faa51b6f1e8dbfd12e/687474703a2f2f7777772e746563686e6970616765732e63666d2f77702d636f6e74656e742f75706c6f6164732f323031342f30342f476f6676c652d4368726f6d652d70726f6365737365732d696e2d57696e646f77732d5461736b2d4d616e616765722e706e67/>