Competition in the Mobile Ecosystem - Browsers & Web Apps

A response to the NTIA request for comment

VERSION 1.0

Open Web Advocacy
contactus@open-web-advocacy.org

1. Table of Contents

2. Introduction

OWA would like to thank the NTIA for soliciting public feedback on the state of the competition in the mobile ecosystem and for the thoughtful nature of the questions asked.

App stores are the primary way that users install apps on mobile devices, and for iOS are the only way for users to install capable apps. Mobile operating systems and hardware gatekeepers exert extreme control over users' devices long after they are sold. The app stores and device restrictions allow these gatekeepers to extract extensive revenue from users **not by merit or user choice** but by **blocking the user from installing software from third party providers** that compete with their own offerings. They can further leverage this power to **charge the user additional fees** (up to 42.8%) on software and services from third party providers.

The software market is highly competitive, it is not possible to add 42.8% fees and not have the **majority** of costs be passed to consumers. OWA views these fees as an unreasonable tax on the user.

> *If a developer for a given number of users requires $7 per a user to support and maintain their product and the gatekeeper demands a 30% cut, in order to receive $7 the developer must increase their price to $10, thus increasing the price the consumer pays by 42.8% (1/(1-.3) = 1.428).*

Various regulatory agencies around the world (including in the US) have proposed forcing the mobile operating system gatekeepers to allow third party app stores. While this would definitely go a significant way to alleviating these issues, app stores **would still wield considerable power** as users would likely only install one or two. Additionally, depending on implementation, Native Apps would still have to be written in different programming languages and with different APIs which are **bespoke** to each operating system. This means that **Native Apps would not be interoperable** between systems and would still require significant cost to build and maintain multiple versions of the same App while providing the gatekeeper an additional form of lock-in via non-transferrable investment and expertise in that ecosystem.

Web Apps are an untapped source of interoperable competition with the app stores.

In a world of capable browsers, Gatekeepers are not able to block their installation and would thus have no leverage to add additional fees or block Apps that compete with their own offerings.

Web Apps are applications installed from the browser that have the potential to **provide users an experience on par with Native apps**. In competent, modern browsers they are interoperable between operating systems, work offline, have a superior security and privacy model. They are, in short, capable of amazing things.

Investment into the Web platform is highly desirable for future competition as it is both open and free and allows any company to build upon it as a platform. Web Apps are increasingly becoming the primary choice for companies on desktop operating systems. Commercial users

now spend greater than 60% of their time within a browser and many companies such as Adobe and Microsoft are heavily invested in adopting interoperable Web Applications (Word, Powerpoint, Photoshop, Illustrator) because they allow these businesses to produce higher quality, maintainable applications which can target multiple platforms with near-zero friction to users and low costs of operation for developers.

Apple's original vision for applications on iOS was Web Apps, and today they still claim Web Apps are a viable alternative to the App Store. Apple CEO Tim Cook made a similar claim in Congressional testimony when he suggested the Web offers a viable alternative distribution channel to the iOS App Store. They also claimed this during a court case in Australia with Epic.

Despite this **Web Apps are not currently a viable competitor** on iOS for three reasons:

1. **Browser Ban**
   Apple's Safari is the only browser on iOS as all other browsers have been effectively banned.

2. **Missing Features**
   Apple has refused to implement key features (some for more than 10 years) that would allow Web Apps to compete with Native Apps from the App Store, both on iOS and in Safari.

3. **Bugs**
   Apple's iOS Safari is buggy and unstable compared with rivals, rendering it unviable as a platform for applications.

iOS is a significant player in the mobile ecosystem landscape with 46% in the US. iOS users enjoy higher average incomes, creating a market that can not be ignored. This goes a long way to **nullifying** Web Apps development cost and interoperability advantages; not only for iOS but also for Android. Companies that cannot imagine using the web to address the wealthiest, most valuable users are forced to accept the costly logic of re-developing their services for each native ecosystem.

We have written about this extensively in our paper "Bringing Competition to Walled Gardens".

Regulators worldwide have acknowledged this anti-competitive behaviour and have proposed legislative remedies to counter it.

The latest revision of the EU's Digital Markets Acts notes:

> *"When gatekeepers operate and impose web browser engines, they are in a position to **determine the functionality and standards that will apply** not only to their own web browsers, **but also to competing web browsers** and, in turn, to **web software applications.**" (emphasis added)*

The UKs Competition and Market Authority have highlighted this accurately and extensively in their mobile ecosystems market study interim report and write (emphasis added):

> *"As a result of the WebKit restriction,* ***there is no competition in browser engines on iOS*** *and* ***Apple effectively dictates the features that browsers on iOS can offer*** *(to the extent that they are governed by the browser engine as opposed to by the UI."*
>
> *"Importantly, due to the WebKit restriction,* ***Apple makes decisions on whether to support features not only for its own browser, but for all browsers on iOS****. This not only restricts competition (as it materially limits the potential for rival browsers to differentiate themselves from Safari on factors such as speed and functionality) but also* ***limits the capability of all browsers on iOS devices, depriving iOS users of useful innovations*** *they might otherwise benefit from."*

They note that Apple has a multiple incentives to hold back Webkit, hindering Web Apps ability to compete with the iOS App store (emphasis added):

> *"First, Apple receives significant revenue from Google by setting Google Search as the default search engine on Safari, and therefore benefits financially from high usage of Safari. Safari has a strong advantage on iOS over other browsers because it is pre-installed and set as the default browser. The WebKit restriction may* ***help to entrench this position by limiting the scope for other browsers on iOS to differentiate themselves from Safari*** *(for example being less able to accelerate the speed of page loading and not being able to display videos in formats not supported by WebKit). As a result, it is less likely that users will choose other browsers over Safari, which in turn secures Apple's revenues from Google."*
>
> *"Second, and as discussed in Competition in the distribution of native apps, Apple generates revenue through its App Store, both by charging developers for access to the App Store and by taking a commission for payments made via Apple IAP.* ***Apple therefore benefits from higher usage of native apps on iOS. By requiring all browsers on iOS to use the WebKit browser engine, Apple is able to exert control over the maximum functionality of all browsers on iOS and, as a consequence, hold up the development and use of web apps.*** *This* ***limits the competitive constraint that web apps pose on native apps, which in turn protects and benefits Apple's App Store revenues****."*

They propose banning Apple from blocking third party competitors from porting their own browsers to iOS, complete with competing engines.

The UK CMA found Apple's security arguments justifying its ban on competitors unconvincing (emphasis added):

> *"Apple's restrictions on competing browser engines: Apple does not permit the use of third-party browser engines within its mobile ecosystem – all browsers are required to use its browser engine, WebKit.* ***We have not identified compelling evidence to date that suggests that, for dedicated browser apps, the potential impacts on competition or consumers from Apple's WebKit restriction are justified on security grounds.*** *We are therefore seeking to assess the merits of a requirement for Apple to allow alternative browser engines on iOS, at least for dedicated browser apps. This could be implemented by requiring Apple to permit third-party browser engines to interoperate with its iOS operating system,* ***subject to those browser engines meeting conditions that would address any risks that might arise from a greater choice of browser engines (for example, complying with appropriate quality and security standards)."***

Apple collected $72.3 billion USD in App Store fees in 2020. While it has not published the costs of App Store review, payment processing, refund handling etc, it has been estimated that the iOS App Store has a nearly 80% profit margin. Industries with healthy competition feature leading firms with profit margins between 5 and 20 percent. This imbalance strongly implies that Apple's removal of functional competition in the App Store and beyond have distorted the mobile phone market for software and services for about half of the US's consumers, and a substantially higher fraction of mobile commerce.

OWA believes that gatekeepers of mobile hardware/operating systems should compete to offer additional services and software to customers on merit and user choice, not by blocking applications that compete with their own or by charging consumers additional fees on applications/services from competitors. OWA proposes a number of remedies (or desired outcomes of remedies) that we believe are vital to restore competition and user choice to the mobile ecosystem.

Only if regulatory or legislative action is taken can the bright future of software competition be saved, not just for the US but for the world. Apple and Google were each started in garages by two developers, we need to ensure this dream holds true for the next generation of entrepreneurs. Competition is the key to unlocking maximum benefit for consumers.

3. Proposed Remedies

In this section we outline goals that remedies should achieve, in priority order, to restore meaningful competition.

We acknowledge that regulatory progress is often long, and that legal text needs to be written in ways that are broad but still anticipate legal challenges and attempts to delay progress by the gatekeepers. It is our hope that by stating exactly what we believe are the solutions to the problems facing competition in the software industry, it may provide some insight into how to construct statutes and regulations so that they can avoid expensive and (more importantly) slow appeals processes by the gatekeepers. An important aim of proposals should be to create clarity for all parties.

From most to least urgent, we hope that market interventions achieve:

- ☐ **1. Third Party Browsers (iOS)**
  Reversing Apple's ban on competing browsers and browser engines by requiring Apple to allow third party browser engines.

- ☐ **2. First Class Web App Support (iOS / Android)**
  Web Apps integrated into Operating Systems to the same level as Native Apps.

- ☐ **3. In-App Browser Abuse Protection (iOS / Android)**
  Preventing In-App Browsers from hijacking a user's choice of browser.

- ☐ **4. Browser/Web App Equality (iOS / Android)**
  Ensuring Browsers and Web Apps are capable by having the same level of access to hardware/software as Native Apps, the gatekeeper's browser or operating system services.

- ☐ **5. App Store Support for Web Apps (iOS / Android / Windows)**
  The ability to submit Web Apps to each of the app stores easily without having to purchase a specific device (i.e. without a Mac).

- ☐ **6. Safari Core Functionality (iOS)**
  Ensure that Apple (as a **primary** gatekeeper) provides core/basic functionality for Web Apps within Safari to enable them to compete against Native Apps.

- ☐ **7. No Chrome Preferencing (Android)**
  Ensure that Google can not unfairly provide any preference to Chrome via licensing agreements or app/operating system functionality.

- ☐ **8. Website Transparency Obligations (All)**
  Ensure Websites have to publish reasons to users for either not supporting a particular browser **or** if prompting them to use another browser.

3.1. Rationale

3.1.1. Third Party Browsers (iOS)

Reversing Apple's ban on competing browsers and browser engines by requiring Apple to allow competitive third party browser engines is key to unlocking competition with Apple's walled garden from an open, safe, and interoperable platform (the web).

Rationale:

The lack of third party browsers on iOS:

- Removes competitive pressure on Apple to invest in their own browser.
- Incentivises Apple to not implement support for web functionalities, in order to protect App Store revenue.
- Provides no incentive to respond to developer needs or ensure that Safari is reliable/not buggy.
- Protects Apple's Google Search revenue but deprives other browsers of that revenue.

Ending this ban will:

- Enable the ability to create functional Web Apps.
- Reduce mobile application development, distribution and maintenance cost for business, fostering innovation and lowering prices for end users.
- Encourage the development of interoperable Web Apps, easing the switch between mobile operating systems, and fostering the emergence of new ones.
- Protect the future against anti-competitive behaviour and remove incentives to stall progress.

IMPORTANT:

Making a browser work on a new platform is a complicated and expensive task for any browser vendor. To ensure competition is restored as quickly as possible, browser vendors such as Google, Microsoft and Mozilla need to be given the **clear signals** by the NTIA and other bodies that a goal of regulation is explicitly to enable such "ports" and to create a clear expectation of Apple to facilitate these ports. Apple must allow responsible browser vendors (For example those that have the "browser" entitlement that Apple provides today) to access **all** system features necessary to bring the best

and most complete browsers that they can. This must include access to currently private APIs that Apple does not provide access to. Apple must not be allowed to obstruct other browser vendors with any onerous requirements.

Apple has given nonsensical security arguments to delay compliance in response to other regulatory bodies. Care in messaging should be taken to ensure the plain meaning of regulatory intent is not subverted by unfounded claims. No responsible party will object to any **reasonable** security policies (backed by convincing evidence) to protect users. Security policies need to balance utility and security, and this should always prioritise the users interest over the Gatekeeper or Developers.

All operating system vendors delegate some of the responsibility regarding security to browsers, including Apple. It is important to note that it is not the gatekeepers sole responsibility to protect users. There will be many instances where providing low level system access to competitors with strong security track records and dedicated security teams is massively in the consumers interest, even though this means delegating protecting the user to these competitors. These competitors can abide by reasonable, narrow-scope, non self-preferencial security/privacy rules backed back by compelling evidence.

3.1.2. First Class Web App Support (iOS / Android)

To create credible competition with gatekeeper app stores and platforms, it isn't enough that competing browsers be given the ability to access increased capabilities when users visit sites in a browser. All modern mobile operating systems also support (closed) mechanisms for "installing" web sites as apps (a.k.a. Web Apps / PWAs).

All browsers on iOS and Android must be granted the ability to create and manage this class of applications. They must also appear fully integrated with the operating system, and provide the user control over their operation **without preference** to gatekeeper controlled and taxed Native Applications. On Android this would mean providing WebAPK minting to third party browsers. Apple may need to provide new APIs for competing browsers to support this use-case, or access to Apple's internal mechanisms for delivering this functionality through Safari today.

Rationale:

First Class Web Apps on Android and iOS will:

- Ensure that Native Apps do not receive any artificial preferential treatment over Web Apps.
- Provide users a method of controlling the permissions (i.e. notifications, bluetooth, etc.) of Web Apps in the same manner they control Native Apps.
- Ensure that the web ecosystem that is interoperable between all devices and free and open can compete with proprietary platforms that only work for specific ecosystems.

- Substantially decrease costs of developing safe, cross-platform apps by providing a standards-based alternative.

- Place pressure on native ecosystems to bring fees in line with value provided to developers, reducing deadweight economic losses. This is accomplished by providing a credible alternative that developers can switch to if the fees are unreasonable.

- Enable companies and developers to build "apps" on the web. To the extent that Web and Native Apps are indistinguishable, it will increase competition and finally make true Apple's claim that Web Apps are suitable substitutes.

3.1.3. In-App Browser Abuse Protection (iOS / Android)

Apps should not be able to undermine a user's choice of browser or otherwise **intercept, monitor or control** a user's web browsing to third-party sites. Apps that wish to act as a browser should explicitly compete *as* browsers. A user's choice of browser along with preferences, security settings and their extensions (for example privacy enhancing extensions) needs to be respected by every app that loads non-collaborating web content (e.g., excluding first-party pages and ads). This should be enforced by the operating system.

The test to apply should be "*what should happen when a user taps a link inside an app?*"

Rationale:

- Improves browser competition by ensuring that apps that act as browsers explicitly compete with other browsers.

- Ensures that Web Apps can not be broken, or prevented from being installed from within a custom In-App Browser.

- Ensures that a user's choice (and thus competition) is respected.

- Improves user privacy by ensuring their preferences, security settings and extensions are respected and by not letting apps intercept user browser traffic.

This is covered in detail in our regulatory submission "*In-App Browsers - Subverting User Privacy, Competition and Choice*".

Regulatory pressure needs to be applied to ensure that operating systems require that apps respect user choice when it comes to browsers.

Developers should also be given explicit ways to mark their content as being only loadable in the user's default browser. Non-conformance with this should be subject to monetary penalties.

3.1.4. Browser/Web App Equality

Browsers and their Web Apps need sufficient operating system access to provide equivalent functionality to gatekeeper's own apps, browsers, system provided functions, and apps distributed via the gatekeeper's online stores. Gatekeepers should also provide full access to hardware APIs, including NFC, Bluetooth, USB, serial, HID, accelerometers, proximity sensors, elevation sensors, temperature sensors, light sensors, hardware buttons, location sensors, and other sensors available to Native Applications.

The aim of this remedy is to remove any artificial restrictions on what browsers or Web Apps can do either via software or hardware, owing to the superior untrusted-by-default security model that modern browsers enforce regarding these capabilities. Operating systems restrictions are therefore redundant when content is loaded in a sufficiently secure, modern browser.

The EU's Digital Markets Act contains interesting language in this regard (emphasis added):

> *"If dual roles are used in a manner that prevents alternative service and hardware providers from having access under equal conditions to the same operating system,* ***hardware or software features that are available or used*** *by the gatekeeper in the provision of its own complementary or supporting services or hardware, this could significantly undermine innovation by such alternative providers, as well as choice for end users. The gatekeepers should, therefore, be required to ensure, free of charge, effective interoperability with, and access for the purposes of interoperability to, the same operating system, hardware or software features that are available or used in the provision of its own complementary and supporting services and hardware. Such access can equally be required by software applications related to the relevant services provided together with or in support of the core platform service in order to effectively develop and provide functionalities interoperable with those provided by gatekeepers. The aim of the obligations is to allow competing third parties to interconnect through interfaces or similar solutions to the respective features as effectively as the gatekeeper's own services or hardware."*
>
> Draft EU Digital Markets Act (11 May 2022) - Page 46

We recommend the US adopt similar language in legislation that would enable browsers and Web Apps to perform any function that either the software or hardware enables. Note that we are not suggesting this be true for every class of program on iOS; just browsers, and only browsers with a strong security track record.

The word **available** is important as there may be important functionality that is "available" either in the hardware or in the software that is not used by the gatekeeper but which they still prevent access to. It's important to open the door for Browsers and Web Apps to access this functionality as this will spur innovation.

3.1.5. App Store Support for Web Apps (iOS / Android)

It should be possible to submit Web Apps to app stores without the developer being required to purchase or own a specific device (i.e. a Mac with Xcode installed). This would need to include both a web-based process to submit the Web App and the removal of any rules preventing Web Apps from being submitted to the App Store.

Gatekeeper's should remove strictly unnecessary steps or requirements from the process of submitting an app.

Rationale:

- Ensures that firms with limited resources do not have to choose between building Native Apps for discovery in stores vs. a website for discovery on the open web.
- Increases the likelihood that developers will choose open technologies that will work across multiple ecosystems, lowering costs and improving competition.
- Fosters investment and expertise into the free, open, and interoperable web ecosystem instead of siloed native ecosystems. This directly lowers costs to businesses and users.
- Reduces the cost of developing for the app stores by not requiring the developer to purchase expensive hardware or maintain expensive, per-target-OS application wrappers.

3.1.6. Safari Core Functionality (iOS)

As an operating system gatekeeper with control over the default browser, Apple has the ability to directly influence the uptake of Web Apps. Due to their ownership and collection of fees from iOS's App Store, Apple has strong incentives to limit the viability of Web Apps as an interoperable and rent-free replacement for native iOS applications which can only be discovered within its rent-extracting App Store.

Although **competition between browsers is the most important and primary driver of functionality**, it's essential that Safari, as the default browser, should be required to implement basic functionality needed for Web Apps to ensure they are competitive including:

- **Install Prompts/Installability**

  At a minimum, Web Apps should be as easy to discover and require as few steps to install as Native Apps. This includes the ability to display a banner and prompt the user to install an app from Safari. As described in *"Bringing Competition to Walled Gardens"* sections 5.4.5 and 5.4.3.1 the Web App install procedure is deeply obscured, whereas this is not the case on any other operating system.

- **Notifications**

  Push Notifications are essential for a wide range of applications

- **FullScreen, Badging, Deep Links, Screen Orientation Lock**

- **Bluetooth and NFC**

  Without these open-standards-based APIs, entire categories of apps cannot be built and developers cannot develop Web Apps that interact with hardware. This **forces** developers to build Native Apps if they require this functionality. Apple has used proprietary APIs over these otherwise interoperable protocols, and made them exclusive to apps from the App Store which is subject to Apple's taxation.

  Apple has not yet published detailed justification of all their reasons on why blocking Web Apps from using Bluetooth and NFC while providing those functionality to Native Apps is justified on security/privacy grounds. This inhibits Web Apps ability to compete with Native Apps and given their concern, Apple's implementation of Core Bluetooth API for Native Apps is surprisingly permissive. Prior to iOS 13, (late 2019) applications did not even need to ask for Bluetooth permission at all. Even now Apple has failed to adequately lockdown Bluetooth for Native Apps.

  It's revealing that Apple is not implementing Web Bluetooth over security/privacy concerns and, in effect, forcing users to download a Native App with far broader powers when they don't appear to have adopted equivalently strong protections within their Native App ecosystem.

Rejecting Web Bluetooth on these grounds is profitable, if nonsensical.

Rationale:

> Apple benefits from poor uptake of Web Apps and can significantly affect how successful they are by being the default installed browser on iOS with significant market share. To prevent this, a minimum set of viable functionality must be made available to all Web Apps across each platform.

3.1.7. No Chrome Preferencing (Android)

Google should not use their control over the operating system to provide unfair preference to their own browser, Chrome, either through the operating system or agreements with partners (i.e. device manufacturers).

This should include:

- **"Google Search App" (a.k.a. "AGA"/"AGSA") Must Respect Default Browser Choice**

  Google forces manufacturers to place the Google Search App on the first (primary) homescreen of most Android devices. This search box drives an *enormous* amount of web traffic and fails to respect user's choice in default browsers, instead causing Chrome to be invoked whenever users tap on search links that would otherwise take users to their default browser.

  This undermines browser competition on Android and is technically unjustifiable. Google can know if the user's default browser is not Chrome and must be made to respect this choice especially with what is arguably the single most valuable browser integration in any program or operating system.

- Mobile Application Distribution Agreement (MADA) should not require Original Equipment Manufacturers (OEMs) to prefer Chrome, as for example requiring Chrome to be the system default and on the first homescreen.

- Users must have a way to disable "In-App Browsers" system-wide and gatekeeper rules should enforce this policy.

  Rationale:

  For healthy browser competition Chrome should win market **share through user choice and merit**, **not control of the operating system**, dark patterns or via contracts with device partners.

3.1.8. Website Transparency Obligations (All)

Gatekeepers such as Apple, Google and Microsoft control high profile websites. If some of these Web Applications do not work in a specific browser, that can cause users to choose another browser. This has been reported many times as an issue by multiple browser vendors, including Opera and Mozilla.

Imagine the fictional scenario where Google decided not to support any browser other than Chrome for Youtube. This would have a dramatic negative effect on browser competition. There are, however, legitimate reasons why some applications can't work in some browsers including serious bugs and missing features.

OWA suggests that where a Gatekeeper's website does not support a browser which has above a 2% market share, they be compelled to publish a document containing detailed reasoning that prevents support of certain engines. A convention could be established for these files to improve their discovery, e.g. a `compat.txt` file hosted on the website root (or in the `./well-known/` directory). However provided, they should include details regarding:

- **Bugs**

  If the browser contains speed, bugs or stability issues that affect a user then the document must contain a detailed description of these bugs with links to bug tickets filed against engine projects.

- **Functionality**

  If the application requires specific functionality that the browser does not support it must provide links to the missing functionality with a rationale why the application/website requires that feature. For example the current version of photoshop does not yet support browsers other than Chrome, but for legitimate reasons.

If a gatekeeper's website wishes to display a "switch to a better browser" dialog then this must also include specifics as to why it is better in the browser they are suggesting which needs to be accurate. In the banner the gatekeeper should provide a link to a user readable description of these reasons.

Gatekeepers must also provide **"try anyway"** links in these UIs to ensure that users of other browsers that *may* be compatible are able to proceed to their applications, even if they appear slightly broken.

It would also be worth considering applying these rules to all of the largest companies rather than just gatekeepers.

These obligations will help to protect competition, by providing transparency to browser vendors/developers and end users.

4. Detailed Breakdown

This section is designed to contain the granular detail about each desired effect with specific points that can be used to determine if the remedy is successful.

- ☐ **1. Third Party Browsers (iOS)**
  - ☐ **1.1. With own Rendering Engine**
    Browser's must be allowed to choose and ship their own rendering engines without restriction to how the rendering engine should work or behave. The rendering engine is critical for pushing web capabilities forward.
  - ☐ **1.2. With own Javascript Engine**
    Browser's must be allowed to ship their own javascript engine or for any other programming language and have access to any hardware functionality required to make the engine performant and secure.
    - ☐ **1.2.1. Including JIT**
      Javascript Engines with JIT (Just in Time Compilation) must be allowed for performance and compatibility reasons.
  - ☐ **1.3. No App Store Restrictions**
    App Store restrictions must be minimal. For competition it's essential that browsers should be able to bring their vision of the web and apps without interference from the operating system except on narrow scope and heavily justified security issues which are in the end users interest. It is important that app stores' rules impose no restrictions related to the user Interface, functionality or technology.
  - ☐ **1.4. Quick Update**
    Browsers need the ability to push updates very quickly and should either be automatically accepted or given maximum priority for App Store review subject to strict service level agreements from the gatekeeper. This is critical for minimising the end user's window of vulnerability (i.e. the length of time an end user is exposed for) when bugs need to be patched.

- [ ] **1.5. Ability to Install Web Apps**
  Browsers need the ability to install Web Apps and for those Web Apps to be deeply integrated into the operating system.

  An installed Web App should be indistinguishable from a Native App.
  It should be technically possible (but not required) for a browser to create a Web App that also has the ability to install other Web Apps to act as app stores.

  - [ ] **1.5.1 No User/Operating System Interaction**
    The browser must be given full control to be able to install and update a Web App by itself without the operating system interrupting the process. Browsers need the ability to install Web Apps without interaction from the user for specific use cases such as syncing Web Apps installed between devices, and when restoring from backups.

- [ ] **2. First Class Web App Support (iOS / Android)**
  Web Apps once installed should have the same level of integration into the operating system as Native Apps.

  This includes:

  - [ ] **2.1. Control over Web App Settings**
    The installed Web App should appear as an app on the Settings.app page, and on each of the privacy menus (where relevant).

    - [ ] **2.1.1. Custom Settings per Browser**
      Each browser can add individual settings per to its installed Web Apps.

    - [ ] **2.1.2. Custom Settings per App**
      Each app can add custom settings to its Settings page.

  - [ ] **2.2. No Double Permission Prompts**
    Only the Web App should need permission to perform an action. For example if the Web App has permission to send notifications and its installing browser app does not, then the Web App can still post a notification.

  - [ ] **2.3. Equivalent Controls/Integration to Native Applications**

    - [ ] **2.3.1 Context Menu on Home Screen**
      The long-hold side menu needs to be supported.

    - [ ] **2.3.2 Integration with VoiceAssistants**
      Browsers and Web Apps should be able to fully integrate with Voice Assistants the same way Native Apps can.

  - [ ] **2.4. Appears in all areas Native Applications are listed**

Browsers and their installed Web Apps should be given sufficient integration into the OS that they can be listed anywhere Native Apps are.

☐ **2.5. Storage Control**
Users should be able to control a Web Apps storage if they want too. Web Apps should appear on the iOS “iPhone Storage” list as separate entities.

☐ **2.5.1 Offload App**
Consideration should be given as to whether Offload App should be implemented for Web Apps.

☐ **2.6. Reliable/Permanent Storage**
The operating system can **not** delete Web App storage even under storage pressure, unless specifically allowed by the browser that installed it **or** by specific user request either via the installing browser or by operating system storage controls such as “Delete App”.

☐ **1.5.2. WebAPK Minting or Equivalent (Android)**
Android must provide other browsers the ability to mint WebAPKs or an equivalent mechanism of installing fully integrated Web Apps.

Longer term, there is room for debate as to whether Google should be allowed to continue having exclusive access to mint WebAPKs from their server infrastructure and whether they should delegate some of this control to competing browser vendors similar to the trust delegation process used by SSL certificate chains.

☐ **1.5.2.1 WebAPK Minting must be fast**
If Google is to retain exclusive control over the minting service and provides no alternative for browsers to be able to install Web Apps locally, they should be responsible for minimising the time taken to install a Web App. Currently there is a delay of several seconds while waiting for Google’s minting server to respond which could adversely affect all Web Apps.

- [ ] **3. In-App Browser Abuse Protection (iOS / Android)**

  - [ ] **3.1. Third-Party Content only open via User's Default Browser**
    Operating systems need to protect users and competition by ensuring that if a user visits a web page that it opens that web page using the user's default browser. The only exception should be for first-party content so that apps can use a WebView to render their own content or from co-operating third parties such as ad providers.

  - [ ] **3.2. User can choose Remote Tab In-App Browser (IAB) or their Default Browser Per App**
    Web content should always use their default browser to render content although a user should be able to choose whether web content should open within the app or whether it should open externally in another browser.

    - [ ] **3.2.1 Default Behaviour**
      The user should have easy control to set the default behaviour of all apps to either open in an IAB or in their default browser.

  - [ ] **3.3. First/Second Party Content Opt-In Mechanism**
    A mechanism to ensure that WebViews can only load first and second party content would need to be implemented to ensure they can strictly only open first and second party content.

  - [ ] **3.4. Only actual Browsers can be a Default Browser**
    It is important for competition that popular apps defer browsing the web to a user's default browser. Only apps that are actual browsers should be able to receive a browser entitlement. For example, popular social media apps and messaging services should **not** be able to receive a browser entitlement.

  - [ ] **3.5. User's must be able to install Web Apps directly from within IABs**
    When a user is browsing the web from within an In-App Browser they must be able to directly install a Web App.

- [ ] **4. Browser/Web App Equality on (iOS / Android)**
  Browser's should be able to access the following functionality. Web Apps should be able to access that functionality as allowed via the Web Apps installing browser.

  - [ ] **4.1. General**
    - [ ] **4.1.1 Hardware**
      Browsers and Web Apps should be able to make general use of **any** hardware the device provides including functions enabled by the hardware but disabled at the software layer.

    - [ ] **4.1.2. Same access as Gatekeeper's Browser**
      Browsers should receive (but not limited too) all the same abilities/the gatekeepers browser receives.

    - [ ] **4.1.3. Same access as Native Apps**
      Browsers should be able to (but not be limited to) use/access any function Native Apps use.

    - [ ] **4.1.4 Same access to System Apps and Functionality**
      Subject to narrow scope security concerns, browsers should have access to (but not be limited to) the same functionality that System Apps use.

  - [ ] **4.2. Communication Protocols**
    - [ ] NFC
    - [ ] Bluetooth
    - [ ] USB
    - [ ] Serial
    - [ ] HID
    - [ ] **Networking**
      - [ ] Low Level Networking (TCP/UDP)
      - [ ] Firewalls
      - [ ] VPNs (including Apple's Private Relay feature)

  - [ ] **4.3. Sensors**
    - [ ] Location/GPS
    - [ ] Accelerometer
    - [ ] Step Counters
    - [ ] Proximity Sensors
    - [ ] Elevation Sensors
    - [ ] Temperature Sensors
    - [ ] Light Sensors
    - [ ] Other

- ☐ **4.4. Health Related Sensors**
  - ☐ Heart Rate
  - ☐ Glucose Monitoring
  - ☐ Other

- ☐ **4.5. Hardware Buttons**
  - ☐ Volume Up/Down
  - ☐ Off/On
  - ☐ Ring/Silent
  - ☐ Back Tap

- ☐ **4.6. Lower Level Functionality**
  - ☐ CPU Functionality
  - ☐ GPU Functionality
  - ☐ Memory Related Functionality
  - ☐ Disk Related Functionality
  - ☐ Security / Cryptographic Related Functionality
  - ☐ Video/Audio/Image Encoding Functionality
  - ☐ Managing Processes (Spawn/Terminate, etc.)

- ☐ **4.7. Telecommunications**
  - ☐ Baseband/Radio/5G/4G Related Hardware Functionality
  - ☐ Telephony Related Functionality (Receiving/Making Calls)
  - ☐ SMS/MMS Related Functionality
  - ☐ Wake from Sleep
  - ☐ Ability to swap about the default phone call application

- ☐ **4.8. Device Management (iCloud Replacement)**
  - ☐ Device Backup
  - ☐ Device Restore
  - ☐ App Data Backup
  - ☐ App Data Restore
  - ☐ **Device Fleet Management**
    Managing fleets of devices, with the ability to apply settings/policies remotely to those devices.
  - ☐ **Locate Devices**
    The ability to run an interoperable device finding and remote management service (i.e. Find your Android/iOS/Windows/Linux/mac devices from the same service)
  - ☐ Remote Lock
  - ☐ Remote Alarm

- ☐ **4.9. Filesystem Access**

- [ ] **4.10. Software Access Equality**

    - [ ] **4.10.1 Media Player/Music Library**
    The ability to interact with or **replace** the system default media player including on the lock screen (i.e. Apple Music).

    - [ ] **4.10.2. Payment Services**
    The ability for browsers to integrate with the default payment service (i.e. Apple Pay).

    - [ ] **4.10.3. Voice Assistants**
    The ability to interact/integrate and replace voice assistants.

    - [ ] **4.10.4. Time Tracking/Focus Management**
    The ability to track application use and manage application time limits and other restrictions.

    - [ ] **4.10.5. Widgets**
    The ability to create and place home screen widgets

    - [ ] **4.10.6. Control Center (iOS)**
    The ability to add icons and widgets to control center

- [ ] **4.11. AirDrop (iOS and Android Equivalent)**
Although not covered by browsers or Web Apps, OWA believes consumers and competition would benefit from an **interoperable standard** for Apple's **AirDrop,** and its Android equivalents.

- [ ] **4.12. AirPlay (iOS and Android Equivalent)**
As above but for Apple's **AirDrop** its Android equivalents.

- [ ] **4.13. Future Devices (Apple / Meta / Google)**
Future devices such as AR/VR glasses and smartwatches should enable the installation of third party browsers and Web Apps in order to prevent yet another gated ecosystem.

- ☐ **5. App Store Support for Web Apps (iOS / Android)**
  - ☐ **5.1 External Web Based Service**
    The app store should accept Web App Submission via a Public Web Based Service with a goal of ensuring specific hardware or operating systems are not required and to enable automated deployment tools that can submit to multiple app stores from different providers reducing the effort and cost for developers.
  - ☐ **5.2 No Specific Operating System or Hardware Required**
    Developers are not required to use any operating system to submit the Web App to the app stores.
    - ☐ No Mac required.
    - ☐ No XCode required.
  - ☐ **5.3 Minimal Steps**
    The steps and complexity of submitting a Web App are reduced to the minimum required. Note that signing up for a developer account on an app store can be a separate process.
- ☐ **6. Safari Core Functionality (iOS)**
  - ☐ **6.1 Install Prompts (Installability)**
    - ☐ **6.1.1 Minimum Bar is Equivalent to Native**
      The install procedure for Web Apps is at least as easy and requires equal or less steps as the procedure for installing Native.
    - ☐ **6.1.2. No Regression in Installation Speed**
      Any changes should not noticeably decrease the speed it takes to install a Web App
    - ☐ **6.1.3 In-App Browser Installation**
      All In-App Browsers must support easy and seamless installation of Web Apps
  - ☐ **6.2 Notifications**
    - ☐ 6.2.1. Must be Free
    - ☐ 6.2.2. Must not Require an Apple Developer Account
    - ☐ 6.2.3. Delivery time equal to Native Apps.
    - ☐ 6.2.4. Users can easily enable/disable them.
    - ☐ 6.2.5. Full support of the Push API specification.
  - ☐ **6.3. Fullscreen** for <canvas> and other non-video content.
  - ☐ **6.4. Badging**
  - ☐ **6.5. Deep Links**

- ☐ **6.6. Screen Orientation Lock**
- ☐ **6.7. Bluetooth**
- ☐ **6.8. NFC**

☐ **7. No Chrome Preferencing (Android)**

- ☐ **7.1 "Google Search App" Must Respect Default Browser.**

- ☐ **7.2 Mobile Application Distribution Agreement (MADA) should not require Original Equipment Manufacturers (OEMs) to prefer Chrome**

☐ **8. Website Transparency Obligations**

- ☐ **8.1 Gatekeepers must have a compat.txt with detailed explanation for browsers with a market share of greater than 2% that the website does not support including:**

  - ☐ **Bugs**
    With links to bug tickets

  - ☐ **Functionality**
    With links to missing functionality

- ☐ **8.2. "Try Anyway"** button must always be available

- ☐ **8.3. "Switch to a browser"** must describe the benefits to the user including in terms of speed, performance / functionality

- ☐ **8.4.Consider applying all large companies**

5. Direct Responses to Questions

Note: NTIA Questions are Listed in Dark Blue

We have attempted to answer all questions that relate to Browsers and Web App competition.

For a more detailed and in depth discussion, please see our paper "Bringing Competition to Walled Gardens".

5.1. Definitions and Statistics

1. How should we measure whether the app ecosystem is competitive?

A key problem in the analysis of app ecosystem competitiveness is the challenge of measuring deadweight losses from anti-competitive behaviour. To counteract this difficulty, we suggest a focus on the structural costs and competition for services necessary to provide essential functionality within an app ecosystem.

For instance, the web ecosystem enjoys healthy competition among many aspects of the stack (all of the various services required to deliver a website or app) and many hosting providers allow developers to create apps on their infrastructure.

Many “CDNs” (Content Delivery Networks) compete on cost and features to optimise delivery of Web Apps. There is broad and diverse competition in DNS and TLS certificate provisioning services. Lastly, there is competition for users between browsers (the platform or “runtime” for Web Apps) and for payment processing services for those apps (e.g., Stripe, PayPal, Trustly, and Google Pay).

App ecosystems that force these services (and many others) to be fused into a single “bundle” by gatekeepers remove competition from the ecosystem. Identifying these bundled services and charting the ways in which interventions will increase competition and deliver lower prices seems like a productive way to consider this measurement challenge.

Item 4 in the detailed breakdown in Section 4 can provide a good starting point to identify these bundled services.

1a. How should the “success” of an app be measured?

Developers have a wide variety of interests and will therefore define success differently based on the positioning of their business in the market. In many cases, “apps” aren’t the goal, but are instead fronts or outlets to *services* that developers maintain across many operating systems and form-factors (device types, desktop/mobile etc). There isn’t a single, clear line between an app and a service, but in the modern era, nearly all commercially important “apps” require functionality that requires an internet connection and the ability to call a provider’s service APIs.

In general, we can think of successive steps in success for a publisher as:

- The ability to produce a front-end ("an app") with intended local and remote functionality without burdensome or unrelated costs.
- Low-friction discovery by users who may be interested in a service, often through competitively-priced ads.
- The ability to monetize services provided to users with competitive transaction overhead / fees.
- Access to tools and techniques that enable profitable services to reach new users over time without gatekeeper interference (beyond core platform security responsibilities).

1c. Does the reported total of the number of apps available at any one time in an app store have bearing on the state of competition among apps or particular categories of apps? (22)

No.

There are plenty of low-effort/low-quality apps that are either unmaintained and in general not of merchantable quality on both the iOS AppStore and on Google Play. This includes apps who rely on an **intrusive** level of advertising or are basic/simple clones of other apps.

Measuring the volume of high-quality apps is difficult, and not automatable. If it were straightforward, presumably Gatekeeper's app stores would include fewer low-quality apps.

2. Are there any important and specific entities (or categories of entities) such that it would be a mistake to omit—or improperly include—them by defining the "mobile app ecosystem" to focus on mobile devices, such as phones and tablets?

From a technical perspective mobile devices such as phones and tablets are not significantly different from any other personal computing device such as a Laptop or Desktop Computer.

Some differences are:

- Their primary input is touch rather than a keyboard/mouse.
- The volume of private data that resides on the device (although in some cases people's Laptop/Desktop computers contain as much or more private data).
- A greater usage of the device for payments (although other devices are still used for payments)
- That it's carried by the person for the majority of the day.

However the key difference is the mobile app ecosystem operating systems are significantly more restricted than desktop operating systems.

Rules should apply without distinction to general purpose consumer computers (including desktop/laptop/tablet/phone/vr headsets) although a special focus is needed on the mobile ecosystems as that is where the majority of the **current** anti-competitive issues are.

If Apple were to release a Macbook with iOS instead of macOS (technically feasible today), the same anti-competitive issues would be in play.

2a. If so, how should this study be scoped so that it is optimal but feasible?

OWA recommends that NTIA study the practical substitutability of Web Apps and Native Apps under the current competition regimes constructed by gatekeeper policies. Apple has claimed in court that Web Apps can be adopted in lieu of its App Store terms and conditions, but developers report that this is (charitably) a misleading claim.

To the extent that Web App substitution requires true competition between browser engines, and fair terms for browser competition, we encourage NTIA's study to include these concerns as a way to emphasise effective, available, and low-cost structural reforms rather than small interventions which may have limited effects in practice.

2b. For example, should mobile apps offered specifically for enterprise use (*e.g.,* for use by businesses, not for consumers) be considered in this study?

Both Google and Apple provide enterprises (businesses that own devices issued to their employees) with alternative policy mechanisms for operating and managing those devices, including the ability to construct custom applications not subject to app store policies.

If NTIA considers these policies, OWA suggests that they be taken as evidence of the broad leeway gatekeepers have in creating tiers of trust that enable certain categories of apps that enjoy different rules of the road. That flexibility will sit at the heart of any new policy regime regarding browsers, and we note that it is not new or unexplored.

3. Apps are not all the same. For example, some have different technical features and capabilities (*e.g.,* location-based apps compared to messaging apps), while others are bound by specific regulatory guardrails (*e.g.,* banking apps or children's apps). In the context of framing competitiveness within the ecosystem, how should we categorize types of apps so that they are grouped by distinguishable barriers and other significant factors? Are there ways to best categorize or segment the market to diagnose specific market barriers, such as those that could impact app developers, or consumers?

To the extent that OWA is concerned about app categorization, we note that Apple (and to a lesser extent Google) have *already* created a specific category of app – the browser – that is special and historically distinct from a policy enforcement point of view.

Instead of applying its long-running policy of removing apps that Apple has viewed as "duplicating functionality" of iOS, Apple has instead *forced* makers of other browsers to deliver fully identical features in many areas of browsing, thanks to Apple's own restriction on browser

engine choice. Yet Apple has, for more than a decade, allowed those browsers to exist and has recently cemented this category with a new entitlement.

Apple thereby admits that browsers are a unique category with special powers relative to other apps. You can see this from Apple's own practices.

We recommend that regulators ratify this category. Browsers *are* special. They require enhanced powers to keep users safe and to deliver advanced functionality as a competing platform / runtime to the Native App platforms that gatekeepers tax developers and service providers to use today. Browser makers intentionally accept this responsibility and should be rewarded for their care.

We are certain that NTIA will consider other categories and factors in its final draft, but insofar as we believe the web can help repair the mobile app ecosystem, browsers must be empowered to be capable enough to deliver competitive apps.

3a. Should distinctions be made based on type of content and app functionality?

OWA is narrowly interested in questions related to Web App substitutability with Native Apps, and the enabling factors related to browser competition that can enable or preclude the web from being a true substitute for native.

To the extent that web browsers are already understood to be available within stores without regard to the content they display or enable, we believe that their existing categorical differences to other apps listed in stores are justified. Should gatekeepers attempt to preclude the distribution of Browsers through app stores on the basis of the content they can display, we believe this would be anti-competitive and adverse to the interests of users and publishers.

3b. Should distinctions be made based on the level of hardware or operating system integration required for the app to function? For example, categories might include apps that access location data, special-purpose hardware (*e.g.,* near field communications), secure elements for payment, or other credentials.

OWA is neutral on questions of functionality in most classes of apps. However, we do believe that the existing, expanded levels of system access that modern browsers enjoy relative to the constraints that Apple (e.g.) places on other apps are justified. To facilitate effective browser competition, safe, sandboxing browsers with a strong track record of security must be granted deep system access to deliver competitive features.

4. How should web apps (browser-based) or other apps that operate on a mobile middleware layer be categorised?

Web Apps run on a middleware layer (the browser) that is relatively unique in the world of software in that a large fraction of the UI, security, and privacy choices related to the applications themselves are subject to choices users can make about their browser, without requiring expensive or difficult replacement of hardware or operating system.

In a pure sense, the web and the practice of effective browser choice is unique, in part, because it represents the pinnacle of US policy choices that promote the legal and regulatory basis for industry standardisation, and through it, interoperability. The emergence of a computing platform that facilitates user choice in platforms/runtimes without large expenditures of money is a shining example of US industrial policy delivering lower costs and consumer benefits.

Apple's iOS and Google's Android arrived into a world already transformed by the web, but have succeeded in closing over a commons that had been tenuously opened through competition and interoperable standards.

This reduction in the ability of browsers to credibly satisfy user and developer needs coincided with early smartphone system constraints, but those constraints no longer dominate. Nearly every smartphone sold in the US contains more than 2GB of RAM, and all iOS devices feature astonishingly fast CPUs. In this context, the browsers and Web Apps could easily provide a strong substitute for Native Apps, as they do on desktop operating systems today, where browsers (including on Apple's macOS and Google's ChromeOS) are able to provide compelling feature-sets that enable developers to provide high-end experiences without jeopardising user safety.

NTIA may wonder if or how other middleware layers should facilitate app delivery, but OWA is agnostic on this point. We only note that browser makers have a unique track record of delivering high security for their middleware, often leading the industry in technologies such as sandboxing, hardware-enabled protections, and transport security. Perhaps other communities could provide similar value, but we believe that it's enough to admit (as Apple does by continuing to ship Safari to its iOS users) that browsers and web are a unique category. We don't need a complete theory of middleware to assert that high-quality browsers are more than safe and capable enough – at least when operating system vendors and gatekeepers do not hobble them.

5. There are some indicators that there is a difference in kind between some apps that generate large amounts of money or are downloaded often and most other apps. For example, one industry analyst reported that 97% of publishers that monetize through the Apple App Store earned less than $1 million per annum in 2021, compared to other reports of more than $1 billion earned by the top 13 apps (including games) on both Apple and Google platforms. (23) What is the best way to assess the competition environment for less popular apps and start-ups?

App stores are heavily centralising and exacerbate “winner take all” dynamics in tech, particularly when coupled with gatekeeper’s highly opinionated and arguably self-serving policies regarding implementation technologies.

Small publishers and startups are least able to afford custom, per-OS development efforts. This helps to ensure that large, well-capitalised firms which can afford multiple development teams enjoy an advantage over smaller firms that may only be able to reach a fraction of users through app stores, or may not be able to maintain high quality apps thanks to the added costs of having to re-develop identical features in multiple disparate technology stacks.

OWA notes that the web has levelled this difference for several decades on traditional PCs, and we believe that if mobile gatekeepers are forced to facilitate true browser choice on iOS and Android, a similar lowering of prices for new entrants and smaller players will come to mobile.

It’s worth dwelling briefly on the question of why mobile gatekeepers have benefited from this oligopolistic situation. Would they not be better suited to richer product catalogues in their app stores? It isn’t clear that they would. Built in barriers to competitive interoperable Apps due to the required duplication of development effort (Apps must be rewritten for each operating system in different programing languages) combined with the ability to block any App from their app store and the ability to give "sweetheart" exceptions to app store rules gives oligopolists additional leverage and rent extraction opportunities. Finally, as these firms may also be potential acquirers of startups in the technology industry, added leverage over them helps their bargaining position when acquiring talent or pricing in potential merger and acquisition scenarios.

5a. Can any potential harms, such as deficiencies in data security and privacy protections, be traced back to the current market imbalance?

Yes. For the Native App ecosystems a user has a choice either to purchase an Android device or an iOS device, from that point they can have marginal control over how the apps operate.

Web Apps by comparison are installed from the user's chosen browser. Every browser by default is significantly more sandboxed and has greater privacy and security than any Native App, requiring individual permissions to access potentially privacy breaching functionality.

Users can use many methods to control security and privacy using the web where no alternative exists in Native Apps.

Native apps do not provide the user any agency in controlling the app. There is no equivalent to browser extensions for Native Apps. User’s can make choices about which browser they want

to use and that facilitates both privacy and security choices that then apply to every app they install. Operating System gatekeepers have not provided any means of achieving this through Native Apps.

5b. Is there evidence to suggest that consumers are less likely to avoid or stop using a particular app even if they would prefer a more privacy enhancing environment because of a lack of competitors offering similar services?

Application developers have perverse incentives to move users to Native Apps and away from the web, even when they are delivering nominally identical services. This is most true regarding advertising and tracking. The set of permissions and capabilities provided **without active user consent** to Native Apps has created a huge market for privacy-invasive mobile ads networks. Those ads platforms can monetize better than web ads (in many cases) because the tracking that underlies the ability to accurately target ads at specific users is much easier in the less privacy-preserving Native App context.

As an example of this, it is worth considering that recent drama over Facebook and Apple tussling over "Ad IDs" relates to a capability **that has never been exposed to the web**, out of (legitimate) fear regarding the use of such an identifier for tracking purposes. That Apple and Facebook are arguing over moving back to a position that is only as privacy-preserving as the web has been signals the gap in potential value to advertisers over this tracking. See also Eric Seufert's two-part series "Apple Robbed the Mob's Bank" [Part 1, Part 2].

5.2. Software and Support for Developers

6. What unique factors, including advantages and obstacles, are there generally for app development — especially start-ups — that are relevant for competition? (24)

Startups face higher hurdles in a world dominated by app stores.

Unlike the previous generation of startups in the "Web 2.0" era, modern startups must factor in **heavy advertising costs** related to user acquisition in the app store. Because the Native App model is more cumbersome (users cannot just click a link, they must download a full application before they can begin to experience a service), strong advantages attach to incumbents that have already convinced users to download their apps.

This winner-take-all pressure exists not only regarding device space and acquisition costs, but also in the ways that app store operators create opaque and heavily favouritism-based "hits". Developers of Native Apps have reported for many years that you need to "know someone" in Cupertino to arrive in the coveted "featured" list within the App Store. For small businesses, this can be make-or-break, and instead of a transparent market that responds efficiently, gatekeepers wield extreme power over small firms that may be potential acquisition targets – or future competitors.

The NTIA will no doubt be aware of many examples of Apple gating publication of apps that compete with Apple services into its store on capricious and opaque grounds. While OWA believes that reform of these publication rules is overdue, we are more interested in **systemic fixes to this entire class of market power imbalance**; namely the ability of browsers to provide effective, capable web-based substitutes for most classes of applications, providing developers with a credible alternative to "knowing a guy" in the San Francisco Bay Area.

The unbridled power of todays' gatekeepers harms users in multiple ways. First, it reduces choice and increases switching costs, driving up potential margins and locking users into single-vendor ecosystems where their transactions are taxed. Next, it reduces data portability. And lastly, because of the policies put in place to render the web less competitive, users suffer the downsizes in security related to a mandated monoculture.

None of these are signs of a healthy market or effective choice.

7. Are there particular obstacles preventing more development from different communities, such as by location/region, ethnicity/race, language, or gender? (27)

Apps on Android and iOS for the app stores are written in different programming languages. This means to have an App work on multiple operating systems often means writing the App several times, this is prohibitively expensive. Web Apps could allow developers to write a single App (one code base) and have it work on all systems. However we do not believe this is currently viable due to rules imposed by Apple.

We have written extensively about why Web Apps are not able to effectively compete on iOS in "Bringing Competition to Walled Gardens".

Making apps that are accessible to people with disabilities is a solved problem on a mature platform like the Web (which is not to say that every web developer is competent at implementing these solutions). Because of the prohibitive cost of making one single-platform app for iOS and another for Android, many organisations choose to use a 'middleware' language such as React Native, which can then generate a version for iOS and one for Android. However, React Native is an immature product, with 1,900 accessibility issues logged by developers but not yet fixed by its vendor, Facebook.

This also negatively affects people with lower income. Android users on average have lower incomes than iOS users. Android devices are typically cheaper.

As iOS users are on average wealthier it is not surprising that they spend more. Developers are driven by revenue and financial constraints. As in many cases developing/maintaining the same Application twice in two or more different programming languages is prohibitively expensive and Web Apps (due to constraint imposed by Apple) are not viable, the developer may choose to develop a single App for a single operating system. As iOS has the wealthy users this is typically the operating system chosen. Many Apps are iOS exclusives as a direct result of these rules.

This is clearly in Apple's favour and a disincentive to either allowing effective competition of third party browsers on iOS or allowing Web Apps to effectively compete with the iOS App Store.

This means less wealthy users on Android are deprived of useful Apps that remain locked in iOS by these barriers.

8. Are there studies or specific examples of the costs or advantages for app developers to build apps for either, or both, of the main operating systems, iOS and Android (which have different requirements)? (28)

8a. What are the challenges specific to multi-platform development and how can they be mitigated?

The primary concern in cross-platform development is access to capabilities and compatibility. To the extent that one platform in a list of supported systems lacks a specific feature, developers can struggle to adapt.

When cross-platform development also includes multiple runtimes, the potential for differing features between those runtimes can also add an additional tax to developers attempting to build from a single codebase.

Web developers have become adept at adapting to many gaps, but there are limits, and additional difficulties caused by a single platform or runtime inevitably add expense to projects.

These are key reasons why Apple's Safari browser is thought of poorly in the web development community today. Apple policy mandates that Safari's WebKit is the only available engine on iOS. Therefore, it is impossible for a developer to recommend another browser to gain

additional features or reduce exposure to showstopping bugs. Combined with the slow pace of feature development, and generally lower quality of Apple's implementation, developers attempting to deliver cross-platform experiences on the web struggle with basics.

The alternative is to submit to the logic (and taxes) of the App Store. This is often vexing, but Apple's low-quality web implementation ensures that developers understand that they will have superior access to features and support using Apple's proprietary tools.

Some developers try to bridge these gaps, using libraries like Cordova or React Native within their App Store-listed Native Apps, but infusing them with small aspects of web attributes (faster development, ability to update out-of-band), however developers understand that Apple may use policy excuses to deny them the ability to publish new versions of their apps if they are discovered to have become too "web-like".

8b. What are the costs and advantages of developing standalone apps for these platforms relative to other means of providing the same services or content, such as web apps, which can operate across platforms?

Web Apps (if the current barriers were removed) offer a number of significant advantages over Native Apps.

Interoperability

For Web Apps interoperability between operating systems becomes almost automatic. Provided the full version of each major browser (complete with their engines) are allowed and they are given sufficient access to the operating system to provide the features that Web Apps need, browsers provide developers a consistent and stable platform to develop on.

Multi-operating system projects can be built with a single code base and do not need to rewrite the App multiple times in different programming languages.

Lower Cost

Native iOS Apps have to be written in Apple created languages (Swift/Objective C) and system APIs that are specific to iOS. You can not run an iOS Application on an Android device (typically written in Java/Kotlin). Web Apps however only have to be written once, in one language and then can run on any device.

In order to support multiple platforms for their application (without using Web Apps) a business will need to produce separate applications for iOS, Android and Windows. This typically involves expanding the team and having multiple code bases effectively multiplying the workload. Keeping multiple code bases in sync is also difficult and time consuming even for large businesses.

This can increase by 2-5 times the development and maintenance cost. These costs must be passed onto consumers. For some smaller businesses it can cause the product not to be produced at all.

Higher Quality

Users also suffer lower quality applications because companies have to split their resources for developing and maintaining applications between two or three platforms, while they could have focused them on only one application.

No additional Gatekeeper Fees

Additionally the business must then pay 15-30% of their revenue to Apple (further increasing costs by up-to another 42.8% for users).

For example, imagine you require $10 per user to cover the costs of developing, publishing and maintaining an Application. The app store that you are selling your App in decides to add a 30% fee. In order to still receive $10, you now need to charge $14.4, which is a 42.8% price increase for the end user. However the actual price increase will be higher as when you increase the price by 42.8% you will lose a percentage of users as you move to a higher position on the demand curve, causing the equilibrium price to be even higher.

Gatekeeper can not block competitors

> *"there's no safety, security or privacy issue - Apple just doesn't like those apps."*
>
> Benedict Evans - Technology Writer

> *"It should not surprise you to know that Apple's interpretation of its text often seems capricious at best and at worst seems like it's motivated by self-dealing."*
>
> Dieter Bohn - The Verge

Gatekeepers like Apple effectively ban certain categories of Applications they don't like or that potentially compete with their own offerings.

No Arbitrary App Store Review Process

> *"there are endless horror stories around curation of the store. Apps are rejected in arbitrary, capricious, irrational and inconsistent ways, often for breaking completely unwritten rules."*
>
> Benedict Evans - Technology Writer

> *"There's a lot of talk about the 30% tax that Apple takes from every app on the App Store. The time tax on their developers to deal with this unfriendly behemoth of a system is just as bad if not worse"*
>
> Samantha John - CEO Hopscotch

The App Store review process can be an extremely stressful and chaotic experience for businesses and developers despite problems with **actually stopping malware.**

9. What role does interoperability play in supporting and advancing a competitive mobile app ecosystem?

Interoperability, as accomplished through web standards and browser competition, has been crucial in improving outcomes for developers building applications on the web aimed at desktop PCs.

Compatibility is somewhat less relevant when critical capabilities are missing. Until a platform can meet the basic needs of a developer, the ability to "port" that experience to other contexts is not a material concern.

Apple scuppers both capability and compatibility, harming the web's ability to lower costs and increase competition on mobile. Apple's Safari and WebKit engine lack key capabilities, driving many apps off the web entirely and into proprietary app stores.

Web Developers whose apps can in theory work on the web even with Safari's constraints often report **extreme** difficulty in achieving interoperability and standards conformance. This adds costs and is a large disincentive to web development. These costs and unpredictability create pressure to move important work into proprietary platforms where it will be better supported by the runtimes.

OWA believes that the best solution to this problem is true browser engine choice on iOS and effective browser choice on all mobile operating systems. True browser competition places competitive pressure on lagging engines otherwise they lose market share through developers recommending alternatives. It also ensures that developers can use tested, cross-platform runtimes that are not dictated by gatekeepers with an incentive to keep the web from competing with the local Native App platform.

Both of these effects are crucial to creating true competition in the mobile app ecosystem.

9a. What are the key characteristics of interoperability as it relates to the mobile app ecosystem?

For users:

1. Being able to install the same app on different devices with different operating systems. This lowers switching costs and creates a larger library of available software.

2. Unified billing in services across devices further reduces switching costs and creates a richer market for quality software.

For developers:

1. Interoperability reduces costs ("write once, test everywhere"), allowing teams to focus on delivering features rather than rewriting them multiple times.

2. Web interoperability creates a gatekeeper-free, safe distribution mechanism that allows developers to publish updates to app functionality and content without requiring costly app review processes. Because Web Apps run on a safer, more mediated/secure middleware runtime, trust decisions can be made dynamically by the underlying browser, which allows browsers to worry less about the dynamic nature of the content.

9b. What other barriers (*e.g.,* legal, technical, market, pricing of interface access such as Application Programing Interfaces [APIs]) exist, if any, in fostering effective interoperability in this ecosystem?

Apple, Google, and Facebook have created unique barriers within their ecosystems to effective interoperability and competition.

As documented in our paper "Bringing Competition to Walled Gardens" and Alex Russell's blog posts on the topic, each of these players has created severe challenges for developers attempting to rely on browsers as their primary mechanism for delivering compelling experiences.

- **Apple**
  Apple's policies against browser competition and antipathy towards web content in its App Store are infamous. Combined with mandated use of an engine that trails the industry on nearly ever metric (compatibility, features, performance, and security), Apple bears chief responsibility for the low prevalence of interoperable software in today's mobile ecosystem. The list of missing APIs and features is too long for us to list here. Please consult our paper for a fuller explanation.

- **Google**
  The Google Search App for Android fundamentally undermines browser choice, harming competing browsers and reducing the potential of developers to trust in the web as a reliable platform. Users that change their default browser away from Chrome will begin to experience the web as a fragmented, amnesiac (different logins / cookies etc) experience. This reduces success for both developers and users as sessions, logins, and permissions are no longer coherent.

- **Facebook (Meta)**
  Meta's apps (including the Facebook apps on iOS and Android) ignore browser choice and instead inject their own (broken) "in app browser". This provides Meta a privileged position for collecting user data, but harms third-party developers and users that may have configured different privacy and security policies.

  Meta's "IABs" lack access to baseline Web App APIs, even on platforms where they would be available (Android).

9c. How are these barriers different or similar than those present in other ecosystems?

No other general purpose operating system other than iOS bans third-party browsers or their engines. Even ChromeOS provides the ability for Mozilla and others to bring their own “real” browsers to users though the Play Store. Every other operating system also allows browsers to provide the ability for Web Apps to be installed as if they were Native Apps.

Lack of functionality in Safari is less critical where alternate browsers are allowed.

Note: Safari is only available on two operating systems (iOS/MacOS) as Apple has chosen not to port it to Windows, Android, ChromeOS, or Linux. Competing browsers such as Chrome, Edge, and Firefox are available on all of those platforms using their own engines.

9d. How does data portability, or lack thereof, factor into consumers keeping the same app if they switch from one operating system (iOS or Android) to another? (29)

This is a significant concern.

If a user has many apps in one native ecosystem, moving to another proprietary, non-interoperable ecosystem may force them to repurchase apps they already owned (due to differences in mandated payment platforms), worry about gaps in app availability, or bear additional costs related to data migration.

Web applications are, generally, a solution to this problem as data is synchronised with cloud services by default. This is what allows transparent access, for example, to one’s webmail across devices, regardless of operating system. Apple and Google have incentive to undermine this model of application development in order **to create higher switching costs** related to their native ecosystems – the surest “upgrade” user is the one already locked into “your” ecosystems, after all.

5.3. Avenues for App Distribution

13. Some mobile apps are pre-loaded on mobile devices or set as default apps, while others are only available through an app store, through a browser (web apps), or, for devices using the Android system, by sideloading. Is there data comparing these mechanisms and their effect on app distribution?

Pre-loaded Apps provide a significant advantage to the gatekeeper as many users will stick with the default provided, however this is far less significant than the gatekeeper being able to either entirely block Apps from third party providers, being able to extract additional fees by blocking the ability to install Apps or being able to block the installation of Apps that compete with their own Apps or services. Additionally we believe that removing barriers to interoperability (for example via Web Apps) is a key method of reducing the ability of operating system gatekeepers from engaging in this style of anti-competitive behaviour.

The EUs Digital Markets Acts latest revision noted:

> *"When gatekeepers operate and impose web browser engines, they are in a position to determine the functionality and standards that will apply not only to their own web browsers, but also to competing web browsers and, in turn, to web software applications."*

The UKs Competition and Market Authority have highlighted this accurately and extensively in their mobile ecosystems market study interim report and write:

> *"As a result of the WebKit restriction, there is no competition in browser engines on iOS and Apple effectively dictates the features that browsers on iOS can offer (to the extent that they are governed by the browser engine as opposed to by the UI ."*

> *“Importantly, due to the WebKit restriction, Apple makes decisions on whether to support features not only for its own browser, but for all browsers on iOS. This not only restricts competition (as it materially limits the potential for rival browsers to differentiate themselves from Safari on factors such as speed and functionality) but also limits the capability of all browsers on iOS devices, depriving iOS users of useful innovations they might otherwise benefit from.”*

Apple's ban on third browser engines allows it to limit the functionality of all browsers and all Web Apps (note browsers other than Safari on iOS and Chrome on Android can not even effectively install Web Apps). We have listed a number of remedies that we believe would restore competition both between browsers and between Web Apps and Native Apps. Additionally this would lead to interoperable Apps that would work on both Android and iOS, lowering consumer costs and reducing lock-in.

As such, while arguments could potentially be made for remedies to address pre-loaded apps and services (i.e choice screens), in the context of browsers we have not included it, as in the absence of **allowing browser engine competition and other remedies, we do not believe this would be effective**.

15. How do, or might, alternative app stores (other than Google Play or the Apple App Store), affect competition in the mobile app ecosystem?

Allowing alternate app stores has significant benefits to end users. It greatly strengthens competition by:

- Reducing how much gatekeepers can charge for access to users.
- Reducing the ability of gatekeepers to block apps that compete with their own.
- Provides direct competition for the app stores themselves without forcing the user to abandon the operating system (and all the apps they have purchased on it) and buy a new phone with a different operating system, or harming their security.

If alternative app stores were available, then the default app store would have to compete with the alternate app stores on:

- Payment processing
- Refunds
- Escrow
- Moderation and content policies
- Discovery
- Reviews
- Developer experience
- Cut of developer revenue

Much of gatekeepers' ability to add additional fees to third parties selling software to consumers comes not from the value of the services they provide (listed above) but from their stranglehold on App installation. Forcing them to compete would greatly limit this, thus in a competitive market will lead to significant reduction in App prices for end consumers.

That said, while this does appear to have considerable advantages to users and significantly reduces the ability of gatekeepers to add additional fees for access to users, this has a number of potential limitations.

Depending on implementation Native Apps are likely still **not interoperable**, as they have to be written in different programming languages and the system APIs are distinct. Users are likely to

install only 1 or 2 app stores meaning the app stores will still wield considerable bargaining power. With higher bargaining power they can extract more money leading to higher costs and less high-quality apps for end users.

Web Apps have a number of distinct advantages. They are automatically fully interoperable. The gatekeepers have no leverage to extract any money from them. They are significantly cheaper to develop as there is only a single code base. For large companies (where development budget is less of an issue) they can achieve higher quality as writing/maintaining the same app multiple times in different programming languages and keeping it in sync across multiple development teams is extremely hard.

15b. What unique barriers are there affecting each of the main operating systems (Android, iOS) that might prevent web apps or—to the extent allowed on Android system—alternative app stores and sideloading, from gaining more popularity with users and app developers than they currently have?

Apple has effectively banned rival browsers from iOS and has made it impossible for third parties to provide the capabilities Web Apps need in order to effectively compete with the iOS App Store. Thus any browser engine functionality missing from iOS Safari is missing from all browsers on iOS.

Apple faces little effective competitive pressure to improve the quality of their iOS Safari browser and has incentives to inhibit it from competing with native. Thus Apple's decade long prohibition on competition for Safari on iOS has a compounding anti-competitive effect as companies sink money into non-interoperable native iOS applications instead of Web Apps.

We write about this extensively in our paper "Bringing Competition to Walled Gardens".

15c. Is there analysis comparing competition on iOS ecosystem (where app distribution is limited) to that of alternative distribution mechanisms on Android operating systems?

While it is possible to install Native Apps on Android directly (rather than through the official app store), it is not possible for users to replace the app store with one from another reputable provider. As such users have to jump through a number of awkward scary security hoops to install each application directly. Unsurprisingly adoption by end users is low (only 1.1% of Android devices had at least one sideloaded app).

Web Apps have the advantage of being interoperable between Android and iOS but that advantage is severely stunted if they are not viable on the other major platform iOS. The previous question and our paper outline in detail our arguments on this.

16. What evidence is there to assess whether an app store model is necessary for mobile devices, instead of the general-purpose model used for desktop computing applications?

The biggest argument against the app store model for mobile ecosystems is the power it allows the gatekeeper to exert over users' devices after they have purchased them both in terms of charging additional fees on every sale of third party software and in terms of blocking (either legally or technically) software that competes with their software services (for example

payment services, app stores, Safari/Webkit, Google Drive, iCloud, Apple Arcade). While there are legitimate security threats that mobile users face, a strong case could be made these limitations are primarily made to enrich the gatekeeper, not to benefit the end user.

Even Apple executives appear to be aware only their stranglehold on iOS installation is allowing their 30% tax on revenue, something they can not achieve on Mac OS.

> *"Neither is on the store because they don't have to be. They can be on Mac and distribute to users without sharing the revenue with us"*
>
> Philip Schiller Apple Upper Management On the Mac App Store

Web Apps are interoperable and run in the secure sandboxed browser environment. We believe that they offer a strong and complementary alternative to the app store model. They are significantly cheaper to write as there is only one code base across multiple operating systems. Gatekeepers have no leverage to charge these apps additional fees for access to users and they can not block apps that compete with their own services. Currently there are a number of barriers that are **blocking this from being viable**. We discuss this extensively in "Bringing Competition to Walled Gardens" and propose a number of remedies.

It is clear from recently uncovered emails showing Apple's upper management's line of thinking on iMessage that they do not see interoperability as desirable and would prefer where possible to lock users into their platform:

> *"The #1 most difficult [reason] to leave the Apple universe app is iMessage ... iMessage amounts to serious lock-in"*
>
> Unnamed Apple Employee

> *"iMessage on Android would simply serve to remove [an] obstacle to iPhone families giving their kids Android phones ... moving iMessage to Android will hurt us more than help us, this email illustrates why."*
>
> Craig Federighi Apple's Senior Vice President of Software Engineering

17. Mobile app stores act as initial screeners and responders for concerns about mobile app content, such as fraudulent apps and malware. (33) Similar issues for screening and responding exist in other contexts, such as website hosting and search engine retrieval. What empirical data is there analyzing any unique content screening issues related to mobile app stores that affect competition?

NTIA is correct in noting that app stores combined multiple, traditionally disaggregated, functions:

- Discovery and search quality
- Developer reputation management
- Application binary delivery
- Payment processing
- Security screening
- Post-hoc security response

In other environments and in traditional desktop operating systems, these functions are either handled by different parties at various stages, or delegated to browser runtimes.

The concentrated power of mobile app store gatekeepers has created a culture of fear regarding many aspects of the "mobile deal". Startups or parties who object to terms of one aspect of the system are hostage to the power wielded by gatekeepers, justified by other bundled aspects.

This is played out repeatedly as leaks, objections, and blog posts documented app store rejections for opaque reasons. But these are anecdotes and not data. NTIA should view the lack of objective data not as a sign that many parties aren't affected, but rather a display of the incredible market power of the oligopolists. We are more than a decade into the consequences of harsh chilling effects.

Nearly every technology business today relies on these gatekeepers continuing to agree to carry their content, in stark contrast with the broad availability of services on the open, interoperable web enjoyed by desktop users.

17a. Is there evidence of legitimate apps being rejected from app stores or otherwise blocked from mobile devices? Is there evidence that this is a common occurrence or happens to significant numbers of apps?

"Legitimate" is the key word in this question. Apple will view anyone who runs afoul of any wide interpretation of its (unreviewable) decisions as being illegitimate. Others will note that these

rules are applied capariously and without consistency and that many large parties enjoy "sweetheart" deals in lieu of lawsuits between similarly armed legal departments.

OWA does not take a view as to which camp is right. We do think that app stores have a right to carry what content they like and edit their inventories as they see fit. Where we agree with Apple less comfortably is in support of Apple's assertion that Web Apps should be a substitute for Native Apps for developers who do not wish to be subject to Apple's rules and regulations.

To make this true in practice, OWA believes that it is necessary to regulate true browser competition into reality on mobile operating systems. Apple, Google, and Facebook have in different ways precluded this possibility from becoming reality, and so there is no true option outside of app stores for developers to fall back on. This harms developers and consumers.

17b. What assessments are there of their effectiveness, or lack therefore, on security and privacy of end users? (34)

App stores propose a model that OWA and browser vendors believe to be particularly weak with regards to privacy and security.

Native App containers and sandboxing systems tend to provide significantly more capability to developers without user consent beyond installation. This gap in consent creates greater opportunities for tracking and the lack of choice in application runtimes reduces the ability of the runtime to be more proactive in moderating the behaviour of developers.

Users on the web, however, can switch their browsers and install extensions to modify negative behaviours of developers.

It is difficult to get an objective sense of app store effectiveness beyond these structural factors, however we will note that malware producers are not particularly impressed with the weak static analysis tools employed by Apple.

17c. Are there disincentives or unique barriers affecting the degree of security and privacy protections offered by alternative app stores?

OWA does not have a strong view on the potential for multiple app stores.

Instead, we favour an outcome in which competing web browsers are empowered to offer installation of Web Applications on fair and non-discriminatory terms. Today, this is not the case. Apple limits the "Add to Homescreen" capability to Safari, and Google fails to provide access to the WebAPK back-end to competitors.

Both parties deny competitors the ability to provide safe, portable, interoperable, heavily-sandboxed Web Applications for easily installation via installable to their own browsers. Opening this feature to competing default browsers may allow for safer alternatives for application distribution than a "wild west" of competing Native App stores which also enjoy access to the overpowered default capabilities of Native Apps.

That is, we imagine a future where “alternative apps stores” need not introduce new or expanded risks due to the insecure nature of Native Apps, but can instead be provided by browsers directly on top of their more isolated, secure and privacy-preserving runtimes.

Even if NTIA agrees with gatekeepers that competing app stores are dangerous, we hope and trust that NTIA will consider browsers to be a separate class, and acknowledge the privacy and security benefits of expanding their role in delivering interoperable software.

19. How does the existence of imposter and other fraudulent apps affect developer incentives or legitimate app lifecycles?

All developers must worry about how to keep their users secure from imposter/fraudulent Apps.

Web Apps rely on the locked down and untrusting nature of the browser environment and correct identification of the website the Web App is installed from using the domain name and https.

Native Apps installed from the app store rely on the vetting process of the app store. Typically this involves an automated code check and a brief click around by an employee of the app store.

App stores (including the iOS App Store) have struggled to keep imposter/fraudulent Apps at bay. Additionally their advertising has lulled users into a false sense of security leading the users to fail to do basic checks such as looking at the App publishers name.

We believe that Web Apps provide a more secure and private mechanism of installing Apps on mobile devices.

5.4. App Users

21. How do most consumers find and make decisions to use apps? Is there data to show whether the usage of an app or any other relevant metric for performance is tied to existing brand visibility outside of the mobile app ecosystem? Is there data about how often people use the search feature in an app store, search engines through browsers, or particular ranking lists of popular apps or app storefronts? Is there empirical data that examines how app rankings, app reviews, or other objective measures of apps (for example, popularity, quality, or number of downloads) are used (or manipulated) to influence consumer choices?

It is very likely that the largest operating system gatekeepers Apple and Google keep detailed aggregate data on whether an App is installed via a link to the app store from the Web (App Clips, App Install Links, Smart App Banners) or via the users directly searching on the app store.

For the purpose of this study it may be useful to **compel** Apple and Google to provide the NTIA with this data. This should include any data they have that involves the interactions between websites and the app store. Additionally third party ad campaign systems for Apps may have useful data they could provide to the NTIA.

The amount of effort that Apple has put into developing App Clips and App Banners (while refusing to implement the equivalent for Web Apps) suggests that App discovery via the Web is material to the App Store's business.

We have written extensively about Web App, App Banners and App Clips in our paper "Bringing Competition to Walled Gardens".

22. The E.O. asks the Department to explore ways to maximize "user benefit" with regard to competition in the mobile app ecosystem. How should we measure or consider user benefit?

22a. What is the appropriate scope of users for consideration? Should it include developers?

Developers should not be considered users in this context. Benefits to consumers should be the primary consideration.

However, in a competitive market, many benefits to developers directly translate into benefits for end users.

For example Web Apps have the potential to drastically reduce costs for multi-platform Apps (i.e the developer only has one code-base and uses the browser environment as a consistent platform). It has the potential to improve quality for large companies as keeping multiple versions of the same App written in different programming languages is difficult and time consuming. Additionally operating system gatekeepers are not able to charge Web Apps 15-30% rent for the privilege of accessing users on their platform. While these are clearly benefits to the developer they translate to higher quality and significantly lower cost Apps for the end user.

If an action benefits developers but on balance harms the interests of end users then in our view that action should not be taken.

22b. If there are conflicts between end-user and developer interests, how does this affect the assessment of user benefit?

In general with careful consideration of the context, end users' interests should prevail over developers interests.

22c. How might convergence of end-users and developers—through low-code environments, for example—affect this dynamic moving forward?

In this context, if those users are providing functionality to other users via low-code they are developers.

23. Do apps that are developed for, or used by, certain communities (such as by income, ethnicity/race, or gender) face significantly different competitive challenges? What are the challenges?

Please see the answer to question 7.

5.5. Other Factors

24. Some apps make use, or would like to make use, of additional mobile device components beyond those that are more commonly accessible (*e.g.,* camera, microphone, contacts) in order to offer an innovative product or service, but the operating system or device provider does not allow such access. (36) Similarly, for some apps, it might be essential to be able to interconnect to other hardware and services, such as cloud services. What are the valid security concerns and technical limitations on what device functionality an app can access?

Gatekeepers of operating systems impose many rules on applications in order to protect user security. Many of these rules are reasonable and in the users best interest. Additionally what is necessary may change over time.

We are not arguing that there be no rules imposed by the operating system on applications that run on it. Rather we are arguing that these rules be necessary, proportional, have strong publicly available evidence and have the user's interest in mind, not the gatekeepers.

24a. What factors should be considered in striking a balance between encouraging companies to ensure proper security measures, while allowing third parties to access the protected features that might allow for further innovation and competition?

The key point here is whether or not there is strong evidence for each security rule that:

- It is narrow scoped and targeted
- It is necessary
- It could not be achieved by less onerous methods
- **The unmitigatable security harms prevented by this rule are significantly and provably worse than the harm to competition caused by the rule**
- It is applied on a consistent basis to the gatekeepers own apps/services and their business partners
- It is not used as a tool to damage competition in favour of the gatekeepers own apps/services and their business partners

Finally it is not the gatekeepers sole responsibility to protect users. There will be many instances where providing low level system access to competitors with strong security track records and dedicated security teams is massively in the consumers interest, even though this means delegating protecting the user to these competitors. Major browser vendors are a prime example of this in relation to features such as creation of apps (Web Apps), process spawning and JIT.

We believe that the onus of proof should be on the gatekeeper to justify these rules and that they must do so publicly and free of charge.

This style of language is already present in both the US's Open App Markets Act and the EU's Digital Markets Act (Section 4).

24b. Are there specific unnecessary (*e.g.,* technical) constraints placed on this ability of app developers to make use of device capabilities, whether by device-makers, service providers or operating system providers, that impact competition?

The Apple App Store Review Guidelines contain the following rule:

> *"2.5.6 Apps that browse the web must use the appropriate WebKit framework and WebKit Javascript."*

Webkit is the engine that powers Safari and several Linux browsers. Apple also produces a "WebKit framework" that is included in its operating systems (macOS, iOS, iPadOS, tvOS, and watchOS.

In practice, Section 2.5.6 is a requirement that iOS and iPadOS browsers from Google, Microsoft, Mozilla, Samsung, Opera cannot use their own engines the way they do everywhere else. These engines take hundreds of thousands of engineer hours to develop, and are excluded from Apple's most successful consumer operating systems. Competing browser vendors are only allowed to produce shells around a very specific, unaltered version of Safari's WebView; a component whose features Apple dictates.

This means that Apple can dictate the feature set of all browsers on iOS and importantly the feature set of all Web Apps on iOS.

In the remedies section we have included a list of legal and technical constraints that we believe need be removed, and a number of obligations that need to be imposed on operating system gatekeepers to restore competition to the mobile app market. Web Apps are an interoperable and rent-free competitor to the app stores that were it allowed to compete would provide significant benefits to consumers and important competitive pressure.

These remedies all have a consistent theme forcing the gatekeeper to allow sufficient access to the operating system to competitors that they can provide services that replace and compete with those provided by the gatekeeper and its app store. In particular allowing both rival browsers and their Web Apps a chance to compete with the gatekeepers browser and app store. This access can be constrained by necessary and evidence based rules for security/privacy/legal reasons as described in 24.a).

24c. Are there other means or factors to consider for mitigating specific risks that would not inhibit competition?

As described in 24.a) there are many such rules that can mitigate common security risks. The key point is they must be justified with strong evidence to prevent gaming by gatekeepers aiming to prevent competition with their own apps and services on platforms they control.

25. What unique challenges, if any, do software updates pose for app competition, including updates driven by the app developers and those necessitated by other ecosystem changes, such as operating system updates? How does this impact security and costs for those apps, products, and services?

Web Browsers on traditional operating systems and websites both enjoy high paces of software delivery versus native mobile apps today.

Based on the lessons learned from decades of remediating security issues, modern web browsers have moved to a “silent auto-update” model in which users frequently encounter new versions of the app simply by restarting it. This enables low latency for browser vendors to deliver fixes for security issues and is one of the key mechanisms underpinning modern cybersecurity (small “patch gaps”).

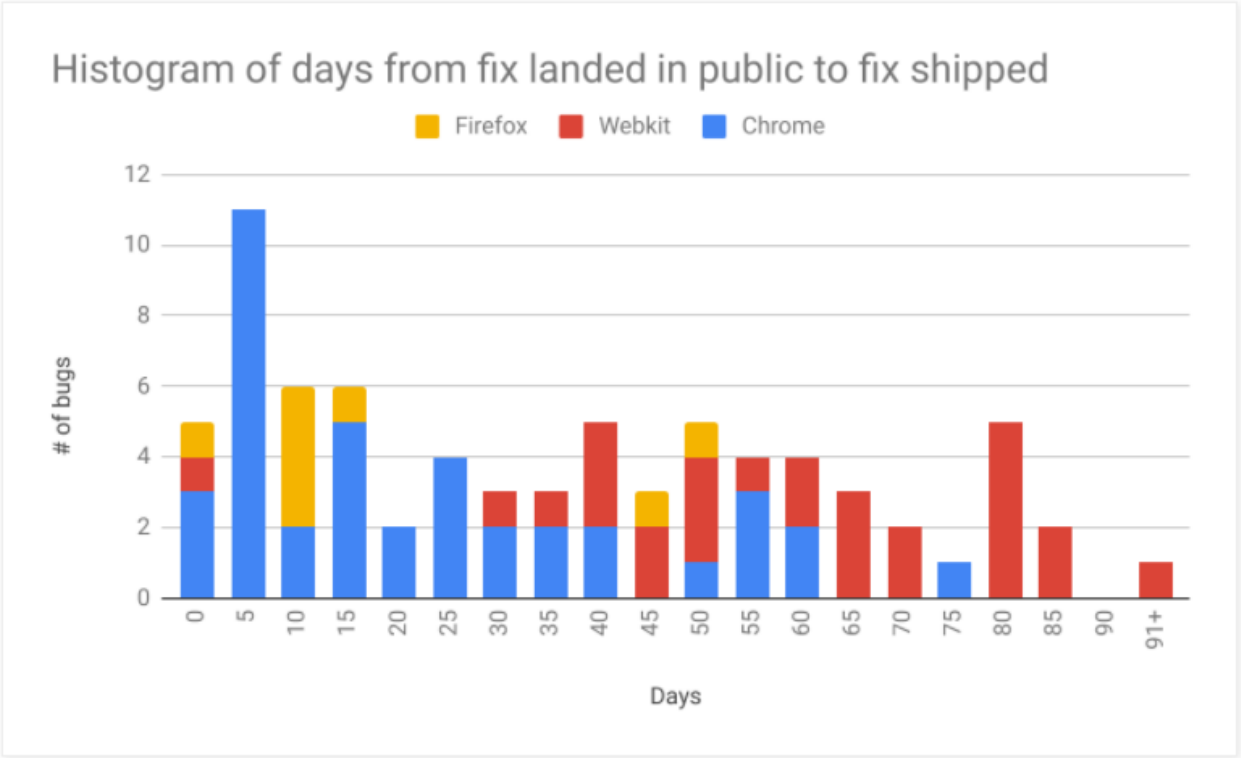
Web developers likewise have enjoyed the ability to update their content quickly and without the intervention of any gatekeepers. This has spawned industry-wide disciplines of A/B testing on live traffic, allowing the entire world of Web Apps to be more agile and lower-risk for developers than traditional software development.

In both regards, mobile operating systems today represent a regression in security and productivity versus the desktop operating systems status quo.

It is important to note here that browser updates on iOS for all browsers are tied to iOS system updates as each iOS update brings with it a specific version of Webkit (the engine that powers iOS Safari) that all browsers on iOS must use without modification. This is an antiquated practice that all other browsers have abandoned due to the issues it causes in delaying users from receiving vital security patches and bug fixes. This means to update the browser, users have to update the entire operating system.

iOS users remain vulnerable to known bugs in Safari longer than users of alternative browsers on every other operating system. This picture is made even darker by operating systems update rates. Since Safari requires a full operating system update, further hassle (and attendant delay) is introduced in getting patches into user’s hands than if the browser updated like a “normal app” (the standard on all other modern operating systems). Safari requires the user to update their entire operating system, a process that makes the device unusable for up-to 20 minutes.

According to the metrics regarding bugs filed by Google’s Project Zero team, Safari is the slowest major engine to fix issues and is significantly slower than others as delivering fixes through software updates.



We have written about the security problem this causes in our paper “Bringing Competition to Walled Gardens” in section “8.2 Security”.

5.6. Potential Actions To Increase Competition

27. What specific measures might the federal government take to foster healthy competition—especially for nascent app innovation—in the mobile app ecosystem?

In the remedies section we have outlined the remedies in priority order that OWA believe are necessary to restore meaningful competition.

We have outlined a brief ratitional for each of the remedies. For a far more in depth and detailed discussion please see our paper "Bringing Competition to Walled Gardens".

28. What specific actions could the private sector and civil society take to ensure and promote healthy app competition (such as technical standards development or monitoring)? (37)

Creating software that is interoperable between devices is incredibly important for both lowering the costs of software development and increasing competition between device manufacturers by allowing software to automatically work on all devices. The Web currently stands as the only viable cross-platform application platform.

Much work has been done on improving the capabilities of Web Apps in the last 5 years. Please see the section “The Web Is Capable” from our paper “Bringing Competition to Walled Gardens”.

There is significant demand for this functionality but in the mobile ecosystem it is being held back by the anti-competitive practices of a few companies. These are gatekeepers who are incentivized to block competition and innovation, not to benefit users but rather to allow them to use their control of the operating system to extract revenue from the software layer.

Were these anti-competitive forces to be removed we believe that the Web and all of the companies involved in it (browser vendors, framework developers, game engines, the wider software community) would invest significantly in improving the Web's ability to provide high quality interoperable Web Apps to consumers.

6. Brighter Future for Mobile Ecosystems

We would like to thank the NDIA again for considering these important issues.

Mobile ecosystem app stores wield a lot of power and lock-in over consumers. This allows them to both block competitors and charge consumers additional fees to install Apps from competitors.

Web Apps could be an interoperable source of competition for app stores however they are not currently viable on iOS for a number of reasons we have outlined in our answers and our paper.

We have proposed a number of remedies that we believe are vital to restore competition and user choice to the mobile ecosystem.

We believe that gatekeepers of mobile hardware/operating systems should compete to offer additional services and software to customers on merit and user choice, not by blocking applications that compete with their own or by charging consumers additional fees on applications/services from competitors.

US regulators have a chance to blunt the anti-competitive power of incumbent gatekeepers and create a level and competitive playing field for the future of mobile App development.

These changes will lead to the following benefits to consumers:

- More competition
- Interoperable Apps
- Higher quality Apps
- Lower cost Apps
- Less lock-in into particular mobile operating systems

Competition not walled gardens leads to the best outcomes for consumers

7. References

Definition of Web Apps
https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps

W3 (Web Standards) Patent Policy
https://www.w3.org/Consortium/Patent-Policy-20200915/

Commercial users now spend greater than 60% of their time within a browser
https://www.microsoft.com/en-us/microsoft-365/blog/2020/08/04/revisit-idea-modern-browser/

Apple claiming Web Apps are a viable alternative to the iOS App Store
https://www.accc.gov.au/system/files/Apple%20Pty%20Limited%20%2810%20February%202021%29.pdf

Tim Cook claiming to congress that Web Apps are a viable alternative to the iOS App Store
https://www.youtube.com/watch?v=H6eYLCxxQdA&t=306s

Apple vs Epic - claiming to congress that Web Apps are a viable alternative to the iOS App Store
https://9to5mac.com/2021/03/25/bypass-the-app-store-says-apple/

Latest version of the EUs Digital Markets Act
https://www.consilium.europa.eu/media/56086/st08722-xx22.pdf

UK Competition and Markets Authority - Mobile Ecosystems Market Study Interim Report
https://www.gov.uk/government/publications/mobile-ecosystems-market-study-interim-report/interim-report

Apple collected $72.3 billion USD in App Store fees in 2020
https://appleinsider.com/articles/21/01/05/app-store-earns-723-billion-in-2020-almost-double-google-play-revenues

iOS App Store has a nearly 80% profit margin
https://www.marketwatch.com/story/how-profitable-is-apples-app-store-even-a-landmark-antitrust-trial-couldnt-tell-us-11622224506

Typical profit margin in competitive markets
https://www.brex.com/blog/what-is-a-good-profit-margin

Alex Russell - Respecting Default Browser Choice
https://infrequently.org/2021/07/hobsons-browser/

Definition of JIT (Just in time compilation)
https://en.wikipedia.org/wiki/Just-in-time_compilation

Apples policy of removing apps that Apple has viewed as “duplicating functionality” of iOS
https://en.wikipedia.org/wiki/IOS_app_approvals#Functional_restrictions

iOS - browser entitlement
https://developer.apple.com/documentation/xcode/preparing-your-app-to-be-the-default-browser-or-email-client

Photoshop ported to the Web
https://web.dev/ps-on-the-web/

Apple Robbed the Mob's Bank (Part 1)
https://mobiledevmemo.com/apple-robbed-the-mobs-bank/

Apple Robbed the Mob's Bank (Part 2)
https://mobiledevmemo.com/apple-robbed-the-mobs-bank-part-2/

React Native has unfixed 1900 accessibility issues
https://github.com/facebook/react-native/projects/15

iOS users on average spend more than Android users
https://www.entrepreneurshiplife.com/why-iphone-users-spend-more-money-than-android-users/

Benedict Evans - Technology Writer - On Apple banning Apps they don't like
https://www.ben-evans.com/benedictevans/2020/8/18/app-stores

Dieter Bohn - The Verge - Apple's self dealing App Store policies
https://www.theverge.com/2020/6/17/21293813/apple-app-store-policies-hey-30-percent-developers-the-trial-by-franz-kafka

Apple ban entire categories of App
https://www.theverge.com/2020/9/18/20912689/apple-cloud-gaming-streaming-xcloud-stadia-app-store-guidelines-rules

Apple ban Apps that compete with their own offerings
https://www.theverge.com/2021/9/16/22676706/apple-watch-swipe-keyboard-flicktype-lawsuit-kosta-eleftheriou

Apple App Store Review is chaotic, irrational and stressful
https://www.ben-evans.com/benedictevans/2020/8/18/app-stores

Apple App Store Review - an unfriendly behemoth of a system
https://twitter.com/samj0hn/status/1431001795904561160

Apple struggles to stop malware on the iOS App Store
https://habr.com/en/post/580272/

iOS App Store is littered with malicious apps
https://www.xanjero.com/news/developer-kosta-eleftheriou-claims-apples-app-store-is-littered-with-malicious-apps/

Alex Russell - Browser Choice Must Matter
https://infrequently.org/series/browser-choice-must-matter/

Only 1.1% of Android devices had at least one sideloaded app
https://www.techtarget.com/searchmobilecomputing/opinion/What-did-2019-see-for-mobile-security-More-Punycode-phishing-and-jailbreaking-returns

Philip Schiller Apple Upper Management On the Mac App Store
https://applescoop.org/story/apple-execs-discuss-why-the-mac-app-store-has-not-been-successful-in-internal-email

Apple Management on why imessage is not interoperable with android
https://www.theverge.com/2021/4/9/22375128/apple-imessage-android-ecosystem-lock-in-epic-games-filings-app-store-dispute

Apple vs Hey (Monopoly Power)
https://www.hey.com/apple/

iOS App Store has a scam app problem
https://www.theverge.com/2021/4/21/22385859/apple-app-store-scams-fraud-review-enforcement-top-grossing-kosta-eleftheriou

Process Spawning definition
https://en.wikipedia.org/wiki/Spawn_(computing)

H.R.5017 - Open App Markets Act
https://www.congress.gov/bill/117th-congress/house-bill/5017/text#:~:text=SEC.%204.%20PROTECTING%20THE%20SECURITY%20AND%20PRIVACY%20OF%20USERS.

iOS App Store - Rule banning third party browser engines
https://developer.apple.com/app-store/review/guidelines/#:~:text=2.5.6%20Apps%20that%20browse%20the%20web%20must%20use%20the%20appropriate%20WebKit%20framework%20and%20WebKit%20Javascript.

Safari is the slowest major engine to fix security issues
https://googleprojectzero.blogspot.com/2022/02/a-walk-through-project-zero-metrics.html

8. Open Web Advocacy

Open Web Advocacy is a loose group of software engineers from all over the world, who work for many different companies who have come together to fight for the future of the open web by providing regulators, legislators and policy makers the intricate technical details that they need to understand the major anti-competitive issues in our industry and potential ways to solve them.

It should be noted that all the authors and reviewers of this document are software engineers and not economists, lawyers or regulatory experts. The aim is to explain the current situation, outline the specific problems, how this affects consumers and suggest potential regulatory remedies.

This is a grassroots effort by software engineers as individuals and not on behalf of their employers or any of the browser vendors.

We are available to regulators, legislators and policy makers for presentations/Q&A and we can provide expert technical analysis on topics in this area.

For those who would like to help or join us in fighting for a free and open future for the web, please contact us at:

Email	contactus@open-web-advocacy.org
Twitter	@OpenWebAdvocacy
Web	https://open-web-advocacy.org