

# Security-Sept 22 Sync

# September 22, 2025 | Web Install API - WP Security Review

**Attendees:** Mike, Dan, Alex, Rob, Lu, Lia, Kristin, Diego

**ChromeStatus:** [Web Install API - Chrome Platform Status](#)

**Goal:** Discuss PEPC and usage of the permission prompt. Agree on a set of restrictions and metrics with which we can proceed to OT.

## Agenda

- Update on internal Google conversations around PEPC and Safe Browsing
- What is the certainty / is there information that helps MSFT be more comfortable with this?
  -
- What are next steps for a PEPC proposal, and who is a primary contact on that team that MSFT can work with?
- Goal
  - what is needed for origin trial?
  - what is needed for ship?
    - work required?

## Notes

- Robert: I see a lot of positives with PEPC - install can go to 'launch', other things - nice and crisp that the user agent can afford being PEPC
  - Current status is 'negative' from firefox, 'opposed' from webkit
  - concern is about a point of time - how many hills to climb here
  - Would like to see us divorce the idea of installing another app from the gating on an element that is 'opposed'
  - In 1-2 years, if PEPC is shipped, then that's great. If it has already shipped -
  - PEPC is not a web standard yet. Building this on PEPC is going to challenge our ability to ship
- Alex:
  - Primary question: risk buckets
    - we understand some fraction of risk for security trust safety
      - This is where PEPC arrives potentially. Can it remove all of the risk that we think we might face? So we don't have to do aggressive cleanup work?
        - my take is no - but if you have data or intuition otherwise, I'd like to know
    - we don't understand other risk
      - we want to have protections to clean things up
  - If the problem is scale - we have lots of tools to help with that
    - examples?

- Mike - **PEPC**
  - You are correct that a permission element is something that neither Mozilla nor Apple has been excited about
  - Recent conversations with Moz about the concept of an element in the page have been productive
    - Shifted from a single element for all permissions
    - to elements that are crafted that are tied to API shape & specific requirements for ability & control
    - expectation - geolocation element agreement will be relatively soon
  - Feeling is that installation mechanism falls closer to geolocation instead of camera & microphone
    - relatively straightforward for interaction model
    - relatively straightforward for what is revealed to page, or what flow the browser is pushed towards
  - suspicion:
    - if we put our heads together, we can put together a crisp explainer to make an element that triggers an installation flow
  - risk for installation
    - PEPC doesn't do much for existential risk for installation. And risk in today's model is relatively low
    - But it does ask questions that the user is not equipped to answer
      - installing things from a different origin is a difficult question to answer given the dialogs that currently exist
    - I feel like y'all can
      - get to a place where the expectations are more clear in the installation dialog
      - ...
    - tying a strong signal intent, that we can provide with an in-page element that communicates intent
      - removes the need for permission
      - gives users more clear context about what installation flow means & intent
- Alex
  - We're on board with that.
  - If y'all have any good information about PEPC - accept vs reject, other ways it triggers on the same surface - data would be great.
- Robert - summarizing
  - We are in good alignment with some of this.
  - Invoking the interaction behavior behind some kind of element interaction. Clicking something, pressing something.
  - Like for us to clear up the permission dialog beyond what we have today? Whatever that dialog looks like..
  - Mike

- We are in pretty broad agreement for first point - I am happier the stronger we think the signal from the user is. A click is ok - but a click is much better if we know that they clicked on text we know about / control etc
  - second point - I'm not asking you to make the permission dialog better. I'm asking you to not have a permission dialog at all.
    - Doesn't make sense to user
    - obviated if we have a signal that the user wants to be in an installation context
    - Then - focus purely on the installation UX flow, making that clear.
  - I'm in favor of removing the permission dialog too.
  - Alex - agree with that - same page with a lot of this.
- Alex - Question - we have customers that would like to do this. We want to validate this with them. We know there is appetite. We want to find a narrow path to run the OT with the current API shape, while we bake something like PEPC. Plan on having something like PEPC assuming that works out. Then also have an API shape like ...[onbeforeinstallprompt? navigator.install?]?
  - So asking if we can ship [as is?] in OT
  - We think that we need something that is API driven, that might be rule-driven
  - We would love to make sure we can open things up much more broadly before more understandable UI over time (PEPC?)
  - Mike
    - Yes - at a really high level, the bit that is important from this meeting - I'm comfortable shifting from an OT, with one thing I want to talk about later.
    - Also - when we are talking about shipping the API as opposed to an OT, I want to press the point that we might not need an API.
      - Alex - let's talk about that later / from a position of success
    - So from an OT, let's make sure we have the data to evaluate
      - obvious: Are people installing stuff, are people getting success business-wide
      - non-obvious: Need to know usage in the wild, and when people are using this in unexpected ways or where it fails or used in poor ways
    - [...keep track of data?...]
    - We need to keep track of the 'install source' for an app - 'installed\_by'?
    - Robert - talked about tracking ... maybe not which origins causing the spike... looking at uptick of apps being installed by other origins?
      - So - if we start to see an increase of apps that are uninstalled, that were installed by another origin, that would be a metric here, that would help
    - Mike - yes - so if you have a plan for what this data is, storing this data, and reviewing this data later, then I'm comfortable with the OT
    - Diego

- Previously we had a way to make a handshake between the installing and installer apps - TAG was very against this
  - Mike - I was talking about something in an email thread at the end of Aug
    - but if we are generally agreeing about the data stored about installed PWAs, so we can recreate the chain if we need to, so we can propagate trust
    - (q: remotely)?
    - Mike - no - if things go poorly, then we would need to integrate with SB that involve a referrer chain
      - but if we do a SB integration in 2027, then current installs are missing
      - so if we ensure that we have that data, so we make sure that we have the data in 2027, then that works
      - Dan: So - locally is good - example - SB gives signal about bad origin X, and we want to know which apps were installed by X.
      - I'm super agnostic about how you do that. As long as it's something we can re-create this data later.
- Robert - The permission element - can you invoke .click on it like anything else?
  - Mike - No
    - Permission element is not something Chrome is going to ship
    - I don't anticipate us shipping in that form
    - I expect we will have specific elements attuned to capability they are associated with
    - That said - they will share infrastructure for making sure user interaction is a strong signal
      - (not allowing .click via JS, no occlusion, etc)
- Alex - we should start an explainer to explore what an element-based installation flow would look like, what API/design options would be in that space, and move forward in parallel?
  - Mike - Yes - and we can talk in TPAC. But I think we can get it done faster.
  - Diego - do you see the install PEPC version a declarative entry point for install API?
    - Mike - that is a way of thinking about it. My personal preference is that we shift the model generally that only shows prompts when they are likely to understand & expect that prompt. Having a control on the page is one way to do that. This would be a declarative version of the API, but also a declarative version that guarantees the UX prior to the flow being triggered.
  - Robert - I'm interested in exploring both declarative and imperative. But not interested in exploring today
    - Mike - then I think we have a pretty clear bar for OT. We can have conversations later. But in the meantime I think we can make progress on

a declarative version here. I would be happy to make a draft explainer, I would be happy for the team building this to be participating in this (from Google)

- Mike - if I'm asking you to do the explainer work, I'm going to commit to helping.
  - [mike can be the primary contact here for reviewing the explainer.md work]
- Mike flipped the security bit for OT to approved!

## Action items

- ☐ Make sure that we store which places apps were 'installed from' (can be multiple).
- ☐ Make sure sereena@ / erinly@ is happy with the new install dialog. (mike: correct but doesn't block OT)

Security/UX-Sept 11 sync

# September 11, 2025 | Web Install API - WP Security Review

**Attendees:** Rob, Lu, Alex, Dan, Serena, Lia

**ChromeStatus:** [Web Install API - Chrome Platform Status](#)

**Goal:** Discuss PEPC and usage of the permission prompt. Agree on a set of restrictions and metrics with which we can proceed to OT.

## Agenda

Decouple PEPC from navigator.install – OT/ship 2 versions

- Usage of the permission prompt for v1
- Why web apps are different than other permissions
  - Little to no evidence of installation prompt abuse
  - Main issues seen around spoofing, addressed via various improvements of features (like updating) to make this less of an issue (<https://bit.ly/predictable-webapp-updating-prd>) - mitigations
  - Permission prompting scenarios
    - User accepts – immediately shown installation prompt, which blocks tab interaction until closed.
    - User declines – permission blocked, no installation prompt.
    - User dismisses – nothing. Site can only reprompt 2 more times before permission is automatically blocked (not specific to web install)
- Any existing evidence of *prompt abuse* mitigation from PEPC?

## NOTES

- Serena
  - I'm supportive of starting with origin trial with existing permission - want to ACK how long you have worked on this.
  - For full shipping - I would like to push back - thinking about UX mostly - I want to make sure that users don't see annoying popups they don't expect.
- Alex
  - Agree on not waning apis going to prompt
  - onbeforeinstallprompt worked because we built in a way no one expected to get it
    - every browser maker who lives with this since hates it
    - 'oh but abuse'...
  - for declarative style API like navigator.install, you need out-of-band abuse mitigation.
    - notifications - we weren't able to expand to abuse mitigation
  - Learning from that - we shouldn't give this to most sites, especially with an API style like this
  - what mitigations should we put in place to clamp down on the set of sites?
  - Proposal



- what if. you can do this if you are installed.
- Or maybe you registered for an OT?
- There maybe are other things we could do.... something like beforeinstallprompt
  - user visits you enough?
- we could do this before PEPC in place, which would let us ship to more sites
- Serena - what is the use-case here?
  - Alex - PWA app stores
  - So - adobe creative cloud, have a whole suite of applications, and they want to have an app-store like page. Similar to what we're aiming for?
    - (alex)
    - Yeah - store.app too. MSFT store. Play store exists on the web. Lots of variants of this you can conceptualize.
    - xcloud streaming - you can go & see a web page, tap to install other PWAs from it that you have paid for.
    - Not opposed to friction to trigger this flow. What level of friction is good?
- Serena - I like avoiding friction in general
  - if we allow installing from a web page, and concerned about cross-origin install...
    - weightier installation flow?
    - Alex - we are looking to you to help us navigate what could work organizationally.
      - Option - try this out behind OT, see potential abuse, mitigate
        - but based on past experiences, that might not be the most credible arrangement.
- Serena - ideal flow?
  - user interaction (no prompting on page load)
  - user gives some intent
    - using a site as an installation hub doesn't seem ... not sure if it sets up the right expectations
  - but we could say
    - you have to click something that says 'install' to launch the install flow
    - you have to have a more involved install flow
    - for first X days, you have to make it super easy to uninstall
    - number of things that could be done to make sure the interaction is expected by user
    - so user can undo the action - another mitigation of that risk.
  - notifications are difficult - difficult to opt-out after opting-in
- Rob - proposal of allowing permission / API only if you are installed too
  - Permission - not sure that a permission is the right answer.
  - I want to always be asked when something is going to install
  - I don't want something to be able to background stuff... I don't want a permission to show up before I install something.
    - So it didn't make a ton of sense to me.

- Serena
  - Permission element is more of a shorthand for **a trigger that has browser control, embedded in the webpage**
  - PEPC - I don't mind if it's a permission we are implementing, but what's important;
    - an element that is embedded in the page, that we can be confident that interacting with it is a user signal
    - e.g. we control the word and the button [INSTALL] (so it's not [GET FREE BITCOIN]).
    - if we have a clear signal that the user is clicking on an element that they understand the result, we don't have to have as much friction
    - thing gives us us problems: **website asks for permission for a thing it might want to do later**
      - or - **I'm going to popup an installation flow now** without trigger / understandable trigger by the user.
  - PEPC has a lot of boundaries and abuse mitigation things in-place & built-in. Main thing, user is triggering the action.
    - Alex - we are on board with using PEPC to trigger this without user friction in the future.
    - question
      - can we do OT without that?
      - What abuse mitig
- Dan – doesn't know if we need to surface to users that this is a permission
  - With PEPC, just use the extra abuse mitigations/guarantees that the user has intention to install
  - PEPC is just the trusted surface/pixels
  - PEPC would make "launch" scenario easier since we control the pixels
    - We aren't revealing to the website if the app is installed or not as the site cannot inspect these pixels
- Lia
  - for permanently blocking - where a site should never allow installation - the permission was a way to do that. But we could do something here maybe custom?
  - Permission prompting scenarios
    - User accepts – immediately shown installation prompt, which blocks tab interaction until closed.
    - User declines – permission blocked, no installation prompt.
    - User dismisses – nothing. Site can only reprompt 2 more times before permission is automatically blocked (not specific to web install)
      - 3 times, it stops asking

- we can do something similar - if our install dialog closes 3 times, we maybe stop showing? If we're not doing permissions?
- Lia - websites requesting permission for something that might happen down the road
  - in the current API design... is this possible?
  - Serena - I can imagine a situation where you allow a site to install another site, and then a week later.... reasoning about what the user might want in the future is hard.
    - allow always?
    - as a user, it's evoking a lot of extra questions and anxiety for a decision made for the future, perhaps for all time. most users don't know how to revoke permissions
    - instead, **if we were to pop that installation flow when you click, and make sure that flow has the information ppl need, feels like a reason installation, then we have less prompts & we communicate what is happening better.**
    - Lia - so you would be more comfortable if there wasn't the permission dialog, and jsut the install prompt?
      - (serena)
      - Yes - but I want more steps. A whole ceremony around it communciates permanence. If we get to a state like that, it communicates what is going on a lot more, sets expectations nicer, and lets them make more accurate decisions and gives them more time to think about if something is going wrong.
      - "What other stuff can this do, I don't know" - with permission
    - Lia - yes we did something in Edge about this, after they accepted the installation.
- Dan - clarification - install dialog always shows up here.
- Alex - Clarification
  - we aren't granting any capabilities for a site that is installed. As rob was aluding to - permission - is just for an origin site to install a destination site by showing the dialog. UI-stuff - whatever we want to do there we're down with.
    - For UI - there is visible state here - taskbar, etc
    - but we grant almost nothing as new features to the installed site.
      - share target is possible
      - request to have your site's storage be durable (no eviction)
      - nothing is added as 'free' here.
        - same as site in a tab here mostly
  - What steps would you want for the heavy door close for installation?
    - How much force from the user do you want in that install flow?
    - Can we come up with experiments and test them?
    - Sereena: We should look at existing flows on OSes
      - I don't think we need all the steps, but usually there is one screen that shows where on your computer it's going to live

- elevation - borrowing some of that stylistically - make it modal maybe? multi-step?
  - If we can communicate that, and pair it with an easy uninstall / undo action, I think we would be pretty good.
  - We're not using PEPC for OT - user gesture? (yes absolutely user gesture)
- OT:
  - user gesture is helpful
- Lu: for OT
  - if we turned up the effort dial for finishing install flow, for information
  - without PEPC, user gesture might be tarnished ([free prize] button) - but we're saying that is acceptable
  - Serena - is this OT 100s of websites?
    - Alex: low dozens
    - If it's trusted actors that we know won't abuse this, then sure that's fine
    - if we're open to handing out OT keys to anyone, then maybe more careful
    - Alex
      - We don't want to use OT as soft launch vehicle
      - If it's only for trusted people, then we won't learn about abuse mitigations we need to put in place
        - OT should even maybe show that
      - and OTs shouldn't languish forever
      - end up with:
        - strict set of criteria we are comfortable with potentially in the long term before PEPC
        - use PEPC when available to reduce initial friction
        - live with both potentially in perpetuity.
      - if that's OK, then what do we need for that high friction?
- Lia
  - what changes are requested here?
    - modal dialog?
    - blur background?
  - uninstall after install
    - IPH bubble sufficient?
      - education with IPH is not the moment they are thinking about it.
    - sereena - personal dislike of those
      - have a more permanent uninstall button in the toolbar that hangs around for a week or so.
    - Lia – show the IPH on every launch for a month after install?
- Dan – what are the hurdles to getting a PEPC element here?
  - UX folks have already been doing research about the install prompt
  - Image with an arrow to show where the app is going
  - Unsure of icon display
  - No capacity until maybe end of December (aka next year)

- People like multistep flows
- Dan pushes back against modal – introduces edge cases, window closing, user ignoring, etc
- Dan likes blurring
- Safebrowsing team wants us to upload information on install and update, Dan started poking around
- What's actually needed for a PEPC implementation?
- Avoid doing all the work for a high friction version only to have PEPC ready by the time we're ready to ship
- [andypaicu@google.com](mailto:andypaicu@google.com)?
- What **exactly** is the diff then between the API and PEPC?
  - Serena: how do we ask questions to users that they have the ability to answer? if we get to that stage, we don't need a lot of friction
    - for anti-abuse - we should have non-UX mitigations.
    - leaning on safebrowsing, making easy to uninstall,
      - safebrowsing - who is the contact?
- Next steps
  - Talk with PEPC - [Andy Paicu](#)
  - Talk with Safebrowsing - TODO dmurph@ and sereena@
  - Talk with Serena & Erin about dialog UX
  - Talk with Mike about OT requirements -

#### Adequate restrictions to the v1 high friction version of the API

- Gating background installation behind the calling site being installed as an app
- Longer running OT?

#### Adequate metrics to collect for the v1 high friction version of the API

- Create UKM for WebApp.WebInstallAPI.Result
  - Also create UKM for sites that prompt users?
- Clarification – How can we tell if web install's one-prompt-per-click leads to a meaningful increase in installation prompt volume?

#### Safe browsing

- Addition of an "installed by" origin field to each web app – Dan to weigh in on feasibility

#### Check in about UXR

- MSFT update – reached out to our UXR folks
- WebApp.InstallConfirmation.CloseReason says it "Records the reason that the WebApp install confirmation dialog is closed. This dialog is shown when the user clicks on the

'install' icon in the omnibox for websites that are installable. **This metric is intended to give data for an experiment to choose the best install icon.**

- [Dan] - any interesting findings from this experiment?

## Action items

- ☐ schedule meeting with PEPC ppl - Andy Paicu Serena Chen Dan Murphy  
(assigned to [liahiscock@microsoft.com](mailto:liahiscock@microsoft.com) )
- ☐ Talk with Safebrowsing - TODO dmurph@ and sereena@ Serena Chen (dan & serena to figure out contact)
- ☐ schedule meeting with MSFT & Serena & Erin about dialog UX (assigned [liahiscock@microsoft.com](mailto:liahiscock@microsoft.com) )
- ☐ Talk with Mike about OT requirements -

Security-Aug 14 initial sync

# August 14, 2025 | Web Install API - WP Security Review

**Attendees:** Diego Gonzalez, Lu Huang, Lia Hiscock, Kristin Lee, Rob Paveza, Dan Murphy, Mike West

**ChromeStatus:** [Web Install API - Chrome Platform Status](#)

**Goal:** Address security, UX, and abuse concerns around the Web Install API, especially on how we can improve the dialog UIs to mitigate security concerns and address any shared sentiments from TAG feedback.

**UX review:** [Web Install UX review - Google Docs](#) | [Web Install API - UX Changes \[383843830\] - Chromium](#)

## Agenda

(Facilitator will be turned on to record meeting notes)

Confirm if these are the main concerns, discuss team responses and questions, and agree on next steps for each:

- Install prompt abuse and annoyance
- Dialog UI lack clarity
- Tag feedback
- (If time permits) What makes this approach right for users today?

---

## Install prompt abuse and annoyance

---

**Concern:** this API could repeat the abuse seen with `chrome.webstore.install()` and `navigator.mozApps.install()`. Currently the API requires transient activation which is too low of a bar.

### Team Response:

- We understand extensions are a very different security risk (different sandbox) than that of websites, but would like more context on `chrome.webstore.install()` and `navigator.mozApps.install()` to better understand the concern.
- [Beforeinstallprompt](#) already shows the install prompt and we haven't seen any real signs of abuse or spam. The Web Install API isn't worsening the risk of abuse.



- Potential mitigation - The UA could gate the background document install capability behind installation. Leaving the option open and clear on the explainer (even though this is mentioned as well) might be a powerful security feature.

#### Next Steps:

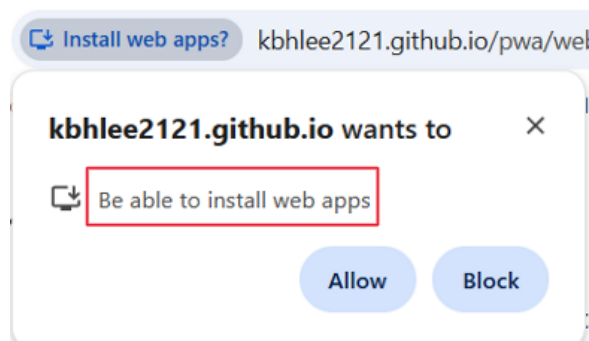
- <Add notes>
- 

## Dialog UI Lacking Clarity

---

### 1. Permission prompt

**Concern:** ("wants to be able to install web apps") is not clear.



#### Team Response:

- Previous review of strings - [Web Install UX review - Google Docs](#)
- Potential ideas
  - UA can screen origins and outright block permission.
  - Alternative strings to start:
    - “Wants to be able to install web apps hosted on another website”
    - “Wants to be able to install web apps that have a different domain”
    - “Wants to be able to install other web apps”
  - Ask the user if they want to revoke the permission if the browser detects that the API has been used too frequently.
  - Don’t require permission at all (Would simplify UX flow).
  - Prevent API usage from sites flagged by safebrowsing (longer-term).

### Question for Mike:

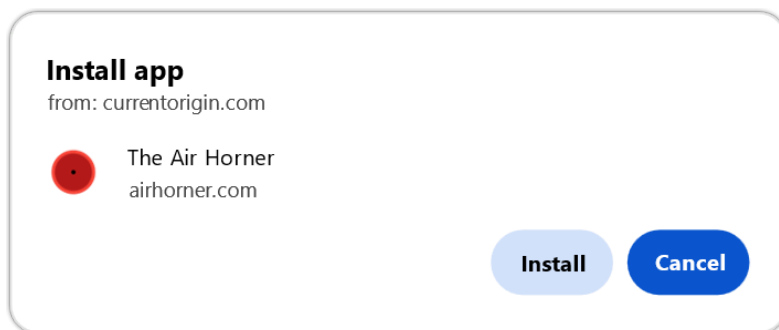
- Can you elaborate on what is unclear? Is it due to users potentially not understanding what web apps are? We previously worked extensively with the Chrome UXW, David Warren, and Dan to come to our current strings, we can reengage with David, but what are the main concerns so we know what to improve?

### Next Steps/Notes:

- Mike: If I see something that is able to install web apps, I feel like every website can install a web app. Suggests to me that people want to install something. Words are hard
  - Load bearing piece of this is the “s” at the end that implies other apps
  - Suggestions:
    - Putting in something about cross origin/other sites.
    - Framing it in the prompt is difficult with its little space which is why prefers to not use it entirely.

## 2. Install Dialog

**Concern:** confusing to have two origins.



### Team Response:

- Chrome already uses this Install dialog for web app installs.
- During UX review, Chrome and Edge teams agreed to add the installing origin:
  - Example: “Install app from play.google.com – Google Maps ([maps.google.com](https://maps.google.com))”
  - [Web Install UX review - Google Docs](#)

- UAs have the option to customize UX (strings, screenshots, etc) and to navigate to the content before install to reduce confusion.
- Proposed mitigations:
  - Alternate strings
    - “Install app - **The current page wants to install** - Google Maps (maps.google.com) ”
    - “Install app - **<current origin> wants to install** - Google Maps (maps.google.com) ”
  - In product help ([IPH](#)) bubble - After install/launch ask users if they intended to install and/or link to uninstall.
  - Add “Learn more” links to dialogs (less likely).

### Next Steps/Notes:

- Mike: reads as installing [currentorigin.com](#) but sees an icon of another app.
  - When there are two origins presented, user’s feelings on one origin likely bleed over into their feelings about the other.
  - Possible for them to ignore the other.
  - It’s a double edge sword - the trust associated with the installing and installee site
  - The site they they’re on and the site that they were installing being different meaning they only saw one of them
  - Tried avoiding the representation of multiple origins. There were APIs that tried this. Payment handler API.

---

## TAG Concerns

---

**Concern:** unresolved concerns from the TAG were pointed out, especially around cross-origin installs. Moving to WICG doesn’t resolve the issues—it just separates them from same-origin concerns.

[\(brand new ✨\) Web Install API · Issue #1051 · w3ctag/design-reviews](#)

[Web Install API - Cross-Origin · Issue #946 · w3ctag/design-reviews](#)

### Team Response:

- Diego is updating the Explainer and adding a FAQ section to address TAG feedback. However, the API has changed so some comments are no longer applicable. **What specifically stood out to you so we can answer the right questions?**
- A few concerns from TAG:
  - The potential centralization and gatekeeping effects of cross-origin installation.
  - The potential for diminished user agency and control over installation.
  - The privacy implications of cross-origin installation signals.
  - "Overall, we'd prefer to see this not go ahead at all...")
    - from Sept 2024 regarding the install\_sources manifest field that has been removed which should mitigate the centralization concern.

#### Next Steps:

- <add note>

---

## Remaining concerns addressed

---

**NOTE:** Responses to other questions raised in the Security feedback. We deemed this as more informational for clarification purposes and can be discussed further asynchronously if we run out of time.

### 1. What makes this approach right for users today?

**Concern:** why is this model appropriate now, given past failures. What has changed to imply different results?

#### Team Response:

- Can you elaborate on what the security concerns are? Are you referring to the extension APIs mentioned above or web apps in general?
- Happy to demo to help show the benefits of this API for developers and users: <https://kbhlee2121.github.io/pwa/web-install/>
- Web app usage is growing.
- Current document - Most browsers already support install flows (even if hidden).
- Developers face friction distributing apps—this API simplifies access.
- Feedback from developers confirms there's a need for smoother install experience.
  - "It's a fantastic improvement, and I really hope this feature makes it to a full public release soon!"

- Install not the current page - it's a new behavior and the UI is somewhat new.

**Next Steps:**

- <add note>

## **2. Clarify financial incentives are not the focus**

**Concern:** financial incentives like the Acquisition Info API. would increase annoyance risk.

**Team Response:**

- Financial incentives are not the core goal of the feature. The core issue the Web Install API addresses is how developers currently use cluttered workaround to get their app installed so this API helps streamline distribution.
- The Acquisition Info API is not implemented in Chromium and is not a web standard. There's been considerations of implementing this in Chromium, but currently we do not know of any plans to do so.

## **3. Chrome developer burden with anti-abuse work?**

**Concern:** who is signing up for Safe Browsing and CPSS integration before shipping on the Chrome side?

**Team Response:**

- Dan's team has plans for Safe Browsing integration in 2026.
- Open to collaboration and demo with the Chrome team.

**Next Steps:**

- <add note>

## **4. What will the Origin Trial actually tell us for strings?**

**Concern:** unclear how we'd get useful information from OT on whether the strings are understandable/effective or not?

**Team Response:**

- The potential mitigations discussed above in [“Dialog UI Lacking Clarity”](#) should hopefully mitigate this.
- Existing histogram data tracks install/cancel rates that can be used as a proxy for whether the dialogs are understood or not. Will need to check with Dan/Vince if we'll have access to this during OT, but the Chrome team will.
- We could look into UXR and lab studies planned to evaluate dialog effectiveness, but it depends on resources and would likely involve support from Dan's team.

#### Next Steps:

- <add note>
- 



## Feature Summary

The Web Install API introduces two capabilities:

- **Current Document Install:** Similar to [beforeinstallprompt](#), allows install of the currently loaded page.
- **Background Document Install:** Enables install of apps not currently loaded (not the current page), gated by origin-level permission.

Installation attempts require **transient activation** followed by a **UA-controlled dialog** showing what is being installed and from where. Only apps with a **manifest** are eligible—no arbitrary content can be installed.

## Notes from Dan

- Install API abuse?
  - doesn't create same problem as extension & notification. Does create opportunities for engagement / reengagements
    - icon in OS. Not audited and sites can change them
    - badging API - allows website bring user attention back to app
    - different in kind from the mechanism that notifications allow w.r.t. engagement
      - but not nothing

- concern a bit - [beforeinstallprompt use-counter](#) - volume is still relatively low. 0.016% of page views. 1/100, 1/50 ish.?? Volume has increase 4x from last year, year and half
  - Whether we have mechanisms in place to determine if that increase in volume is a good thing or a bad thing.
  - Notifications - we didn't have a way to ratchet down on this if bad things happen
- Question - what do we have in place which helps us understand the usage in the wild, helps us understand if increase of that usage is good or bad for users, and then how do we protect against bad stuff
  -

Unclear how many of those prompt resulted in good user activity:

- A few future plans of Dan's team:
  - Safe browsing and existing protection today. Acknowledged not perfect. Reality today is that existing protection today is that all of this is impossible if the site is blocked by safe Browser.
  - Update the Crawler to make it easy to emulate and install PWA
  - UX research: how users treat the word "install". Install currently seen as a scary thing

Mike: Easier to get in front of a problem than dig yourself out. A little hesitant that we are doing a safebrowsing integration later.

- Cross-origin mechanisms as part of this api - make safebrowsing more interesting.
  - installer + installee
  - if we act on an installer, that is more interesting. Installer installing the most malicious applications.
  - hard to do retroactively
  - Thinking about this integration ahead of time, interacting with safebrowsing, seems necessary to launch safely
- MSFT - side
  - since safebrowsing integration - since it's only easy for Chrome folks to do this easily - have you thought about smartscreen integration?
    - Am I the only one raising concerns? Or have others internally raised these concerns?
- Diego
  - Haven't worked with that team in particular. API has had different shapes. Reviewed this with a bunch of people - I don't think this concern has come up.
    - The browser is already doing..... if you're trying to navigate a page that is reported to be unsafe, the browser will block it

- but I don't think we've considered how this integrates with the smart screen. As far as I understand, this is something the browser is already providing, and the browser should be doing smart-screen or safebrowsing that the user agent has
- Mike: The kind of abuse I am concerned about is unlikely to get on safebrowsing's radar. Challenging to get them to pay attention to abuse that is annoying vs abuse that has a meaningful impact on data flow or resources. SB pays a lot of attention to phishing pages, a lot of attention to unwanted software. Do not pay a lot of attention to unwanted notifications.
  - Did a lot of work to define a set of characteristics of unwanted notifications.
  - Not something that SB got us, or the central SB team was going to spend time and energy on.
    - Hopeful, Dan, that they will be interested in spending time on it in this case
    - A little bit...
      - The comments earlier about malicious sites being blocked anyways
      - requires a definition of malicious that is expansive - user surprise vs user harm.
      - Dan:
        - (and SB is mostly focused on user harm)
        - (so not guaranteed)
    - Based on feedback from users in the past - annoyance & noisiness of the web is as much part of the problem as abuse on the web.
  - The reason developers like to have users install a PWA is a two edged sword
  - I want us to think about this in a way that is sustainable
  - I like PWAs. Like install.
  - Worried that we are creating more incentives for developers to pop up install annoyance to users over time. There will be abuse, and there will be annoyance. Distinguishing between the two is difficult.

Diego: How would the process go? Given we have different processes between each company?

- Redefining what malicious can be... is there any way that we can apply techniques to see if the API is being unnecessarily called?
- For development of a feature like this.... if MSFT wants to talk to SafeBrowsing team - how would that work?

Mike: Two thoughts

- Not convinced that SB or smartscreen is going to prioritize this. Have had difficulty in the past defining these annoyances, that require policies to be in place before blocking behavior. Smartscreen - don't know. But SB needs solid policies for many reasons, and it's difficult to define these for the 'user annoyances'.



- In the process of building a chrome-side abuse team. More ability to use signals that safebrowsing is able to provide. But more in the Chrome branding instead of safebrowsing
  - e.g. suppress prompts for geolocation, notifications, etc where user isn't going to gain value.
  - Want to explore how we integration new kinds of prompts here
  - CPSS - chrome permission suggestion servec
    - maybe we can chat with this team to see if this kind of integration might be supported or not
- thought one:
  - Code has landed in Chromium & Edge. You have all the capability in the world to ship things in edge in whatever for you like. You don't need me to approve anything to start experimenting. One answer - start experimenting if that's something you & your security teams are comfortable with.
- other thought:
  - landed & behind a flag. process we have is to do a security & privacy review as process of origin trial. A little concerned about releasing this as user-facing that sites are not opting into. With little understanding of how these things are being used. Sounds like Dan has collected metrics & will continue to collect metrics in the usage of the API
  - Interested in:
    - For existing collection metrics
    - UMA metrics - raw count of how often a thing happens
    - UKM metrics - users who opt-in, we key these counts to the top-level origin.
      - e.g. you know that the install happend from superamazingwebstore.com
      - Doesn't give us insight into a given thing being installed.
      - policy: The metrics that are collected in the context of a page are keyed to that page. Don't generally have metrics on pages that the user isn't actually seeing
        - possible to stretch that a bit, if we have an installation metric that is tied to a thing being actually installed
        - BUT we can't tie those together - the installer & installee.
        - We could see installer is called X times, installee installed X times, but no relationship
    - Establishing how we can record this metric - installee + installer - we should think about this

Mike: BIP decision to remove the engagement

- When we make it easy for devs to put prompts in front of people, they generally do so. I'm curious --- we used to have heuristics in place which would limit the ability to trigger an install prompt to a smaller set of pages than the entire world.
- removing that - Could that have increased the set of sites that could install?

- Any metrics? or entirely unnecessary?
- Found 0 correlation with engagement score for whether the user would want to install a site.
- Engagement score ended up being a weird signal. Needed user to click twice on the page.
- Hypothesis: I can use this now because this is actually more reliable
- ML model got pretty good at predicting install.
- Install prompt acceptance rate is 30%.
- With ML model, achieved similar rate.
- Ongoing UX efforts to improve the install.

Mike - UX:

- Some of the UX seems confusing. Are there options we could explore so it is less confusing?
  - Exploring something now - called "PEPC": [PEPC/explainer.md at main · WICG/PEPC](#)
  - button on page whose content is controlled by the browser
  - we can treat it as relatively strong signal of user intent
  - Signal of attention, but not necessarily of what user wants to do with that attention
  - Exploring this now with
    - search - geolocation
    - meet & other VC products - camera & microphone
  - Nice because this fits in general flow the user....
    - user understands what happens when clicked
    - This in-page UX has a meaning that is more substantial than random pixels we don't install
  - I think using PEPC would obviate the concerns here for install / cross-origin install
    - Feels like a better direction that we should be exploring vs the interaction with what the user THINKS they are doing in the page vs what prompts the browser puts in front of them.
    - With same-origin and cross-origin permissions
    - .... seems reasonable to me that a well-meaning / well-behaved developer will put an install button on the page. It will be following the status quo, use `beforeinstallprompt`, or call `navigate.install` on click
    - It seems like it would be meaningful for us to explore making an install element vs (or with) and install API
    - **An install element whose pixels are understood by the user gives us a much stronger signal that the user wants to install, and mitigates a lot of the risk that the user is surprised that an install element / dialog will be popped up in front of them.**
- Are you all familiar with that explanation? If so, could it fit into this proposal?

Lu:

- I've looked into PEPC a bit? Is it shipped? Is it a reliable mechanism we can rely on?
- Mike: origin trial right now, stay there until there is something specific we are shipping.
  - Meet, zoom right now
  - We would ship that, but mozilla has just started re-engaging, and they have different ideas about how the element may be shaped.
    - Likely we will split up the single element into multiple elements that will be specific to the use-case.
  - Bluntly - permission element is going to stick around, and is behavior that is valuable to users
    - origin trial right now as we're not sure about the 'shape' of it
  - Meet is using today, zoom is experimenting but might not ship yet as it would blow past page view limit for origin trial
  - ppl responsible for permission in Chrome are enthusiastic about moving towards that model
- I like that model - user convenience, and the something on the page is something we control nicely.
- Shape of PEPC and html tag might change?
  - Mike: I wouldn't say that I require it as a security reviewer.... but with my permission-team-adjacent-hat-on - I hope you would explore this concept
    - The thing it could resolve out of my feedback - merely waiting for a click on a page is not an effectively boundary for abuse
    - For FF and Safari will only prompt for geolocation and notifications in the presence of transient activation. But that has not done a whole lot for the prompt rates that they see.
      - Getting a click if you really want one is not very difficult
    - So - not a blocker, but substantially more robust.
    -
  - ALSO MEANS you don't need a persistent permission on the page! That model would put the initial click into the page context, but it's trusted context that allows us to extract a useful user signal.
- Lu: So isn't a reliable signal today that users want to see a prompt.
- Mike - if we can build something that doesn't need us to do crazy analysis for user intent, that would be great.

Mike: What I'll do is send documentation that is currently being produced for PEPC mechanism in OT today, but also new variance like the geolocation element. Geolocation seems like a 1:1 match of what you want to do here. I think it'll be relevant, and I think you should talk to that team. I can put you in touch with them.

Mike: For dialogs that are in the browser:

- The surprise that I had with the first dialog, persistent permission to be able to install applications to other sites.
- .... with the knowledge that I know, which isn't useful about other users knowledge...

- With the knowledge - I feel like every web app can install itself. The kind of installs that this site can do, vs the installs that sites can do for themselves.
  - So - ppl who THINK that the prompt means that the site can install itself, will say "Yes of course, I want you to be installed" without knowing it can install other sites
  - Another example of confused users "Do you want this website to use fonts" - yes of course
- Important part - can install **other** apps.
- I have suggestions. They would help me, but not sure they would help other users
  - Cross-origin
  - other sites
  - want to make it clear the thing this site wants to do is be an app store / app catalog / install other things
- I think we have gone way over-board with prompts in general - it's the blade we have and we use it too much. Not sure if we can meaningfully explain this in 7 words.


When I see "install app from currentorigin.com" that means to me that I am installing that origin. Then, when I see another origin for another thing - that's surprising.

- We have generally moved away from showing more than one origin in a dialog. When an iframe asks for permission, we attribute that to the top level, and we require top level to delegate that ability to the iframe in the first place
  - user can make reasonable decisions about the site they are trying to visit. But cannot make decisions about sites that are not the site I'm on
- Interesting in this multi-origin case.
  - user feelings about one origin might overrule / bleed into feeling about the other.
  - trustedwebstore.com - I trust that.
    - airhorner.com - I don't know anything about that
  - Idea is that the store is doing good work for trusted sites. But that's a double edged sword - they might install sites they might not have done otherwise.
  - The ability to install applications..... (extensions).... because it was valuable to allow ppl to install extensions... we saw a lot of abuse though.
    - When we pushed users BACK to the web store, instead of allowing random install (where users now got application title, all of the trusted content in the web.store site, etc)
      - Abuse rate dropped substantially. Now they only see one origin at a time.
- Anyways
  - origins are confusing to me in this UX - what site am I installing?
  - multiple origins are confusing in general
    - other things that do this mitigate this - FedCM, <other>
    - This presentation didn't help me come to the conclusion that this is mitigated here



# Privacy Review

## WP Privacy Review

 **Comment posted** on feature entry:  
[Web Install API](#)

**A comment was added to this feature:**

**tov@google.com**  
Hey Lia, apologies for the delay. Re 1): I actually meant "PWA" instead of "IWA" in my question. So my question was whether the installed PWA would be able to determine its own app id? I presume that these id's could be random and unique values, which would provide the PWA with a persistent identifier. Re 2): I checked out the demo, and it seems that in Incognito mode the DataError cases show the popup instead of raising a DataError immediately. This would still allow one to distinguish between regular and Incognito mode. Is it possible to perform this check after verifying the request is valid? Similarly, one could use a unique URL as manifest, and afterwards check whether a request was made to it. Re shared concerns: the two main concerns that I share with Mike are the cross-origin installation capabilities, as well as the unclarity of the permission dialog, i.e., that it is unclear what the prompt is asking of users and the consequences this might entail. Feel free to loop me in when discussing these concerns.

### 1) Installed PWAs can know their own app id.

Yes, installed PWAs can determine their own app id, but it's an id that is already publicly available. Apps should specify their id in their site's manifest.json file. If they don't, the site's start\_url is used instead. Per the manifest spec [1], *"The manifest's id member is a string that represents the identity for the application. The identity takes the form of a URL, which is same origin as the start URL."* This id is listed in the Application tab of the developer tools.

The notion of random/unique values is only for internal browser code, and the random values are never surfaced to developers, so apps cannot know them. The manifest\_id returned by the API is the same as the public id field found in the developer tools.

[1] [Web Application Manifest](#)

**Identity**

Name	Financial Times
Short name	FT
Description	
Computed App ID	<a href="https://app.ft.com/?standalone">https://app.ft.com/?standalone</a> ⓘ <a href="#">Learn more</a>

Note: id is not specified in the manifest, start\_url is used instead. To specify an App ID that matches the current identity, set the id field to `/?standalone` ⓘ.

[FT](#) (no id in manifest)

<b>Identity</b>	
Name	Web Install Sample
Short name	
Description	
Computed App ID	<a href="https://kbhlee2121.github.io/WebInstallSample">https://kbhlee2121.github.io/WebInstallSample</a> ⓘ <a href="#">Learn more</a>

[Web Install Sample](#) (has id in manifest)

## 2.1) Server requests for a unique manifest URL

To ensure we're understanding correctly, can you confirm if this is what you're referring to:

- Concern: Bad app developers can use the presence or absence of a server request for the manifest id url to know if a user is in Incognito or not.
- Flow examples:
  - Bad installer and good installee - User clicks "install example.com" on badsite.com -> example.com sees a server request was made for their site, but doesn't know from where.
  - Bad installer and bad installee - User clicks "install badsiteA.com" on badsiteB.com -> badsiteA.com sees a server request made for their site -> badsiteA.com must work with badsiteB.com to know when to expect a server request from navigator.install.
  - Bad installer installs itself (0 param version) - User clicks "install badsite.com" on badsite.com -> badsite.com sees a server request. If they don't see a server request when they expect, what info does that infer?

If yes, that is a great point. Here's our response:

- The site being installed (the installee) can know if the server made a request for their site, but unless the installer and installee have the same origin or are working with each other (the installee asks the installer to use a unique manifest URL for Web Install such as "example.com/fromwebinstallapi.html", the installee would not be able to know if a request was made from the Web Install API or not.
- Additionally, the absence of a server request is not indicative of anything specific and could imply several cases. In addition to Incognito mode, it could also mean
  - The user didn't click on anything that invokes navigator.install
  - ISP dropped packets
  - The browser crashed
  - Network issues
  - The user blocked the web install permission
- Overall, this is a good point about the server requests, but we don't see this information alone being indicative enough to app developers of whether a user is in Incognito or not. If there's something we're missing, please do share.



## 2.2) Data error / Incognito

Can you also help confirm if we understand this correctly here?

- Concern: An app developer may be able to determine if a user is in Incognito mode from the different result (AbortError vs DataError) returned from the same navigator.install element on the page.
- Flow: badsite.com deliberately has an "install (return DataError)" button that they know will fail the manifest id validation
  - User clicks on the button in a regular profile -> DataError is returned immediately.
  - User clicks on the button in Incognito -> the Install Not Supported Dialog shows and AbortError is returned.

If yes, this is another great point. Here's our response:

- We see the concern of bad sites inferring Incognito mode upon receiving a AbortError when they're expecting a DataError. It's true that AbortError is returned when a user is in an off-the-record profile, but there are other possible cases when AbortError is returned (without a server fetch too) such as if the browser crashed, if there was a network issue, or if the user denies, or previously denied, the web install permission.
- Essentially, we don't immediately see how this information is indicative enough to conclude a user is in Incognito, because it could also mean the other possible scenarios mentioned above. Let us know if we're missing anything here too!

3) Re. shared concerns on cross origin installation capabilities and permission dialog - We've added you to the email thread with Mike on these topics. Unfortunately, Mike is OOO until mid September, but do feel free to chime in there.

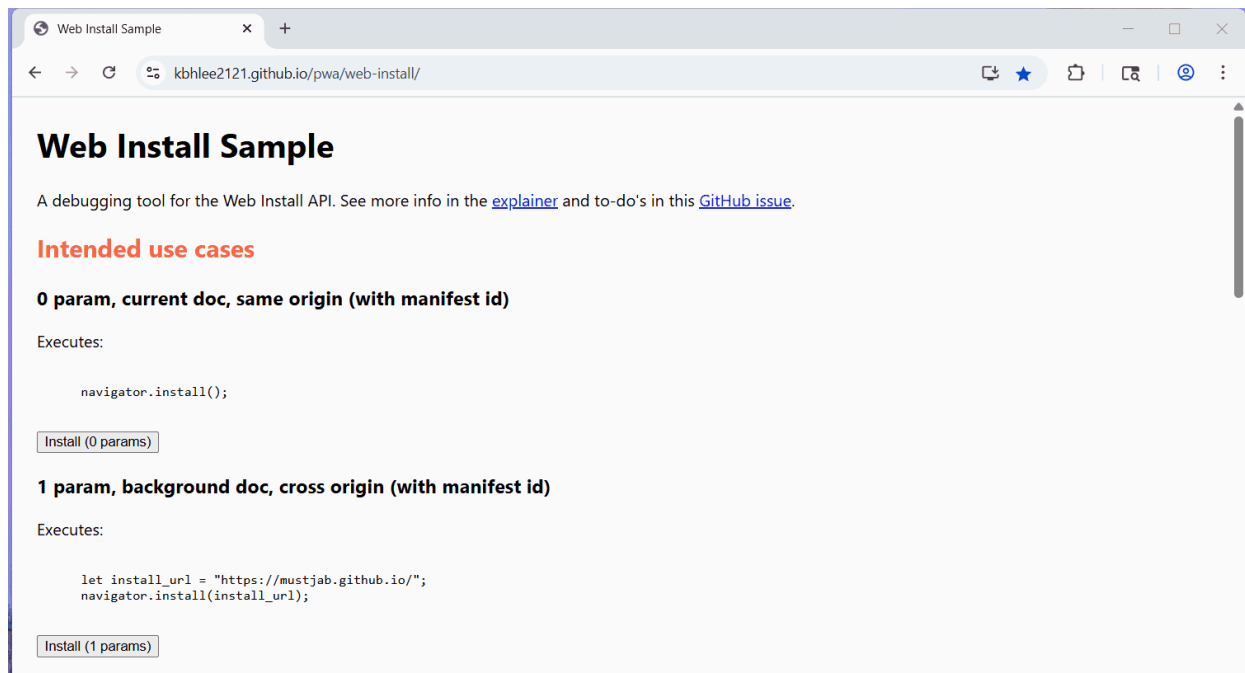
# Roadmap to OT

# Web Install Roadmap to OT (and beyond)

Last updated: Sep 29, 2025

[Explainer](#) ~ [Cr bug](#) ~ [ChromeStatus entry](#) ~ [UX Review Doc](#) ~ [Test Site](#)

## Current UX

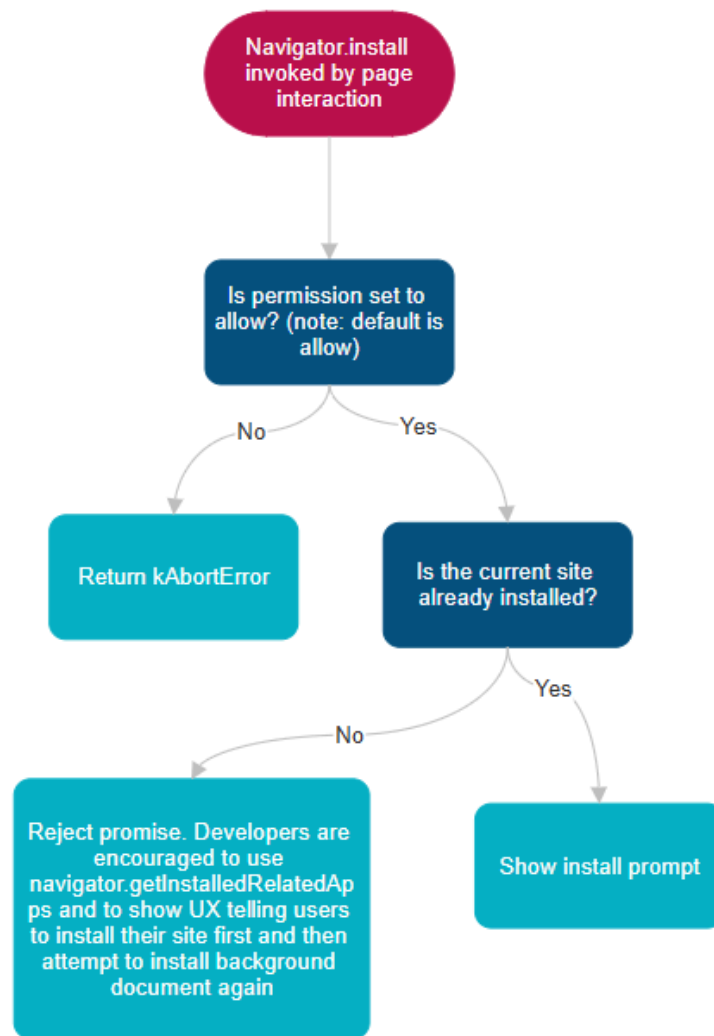


background-install-full.gif

## Proposed plan for OT (background/cross-origin installs)

(Now) Proceed to OT with a high friction, imperative version of the API

Require a site to be installed as a PWA to use the argument version of the API (background/cross-origin installs).



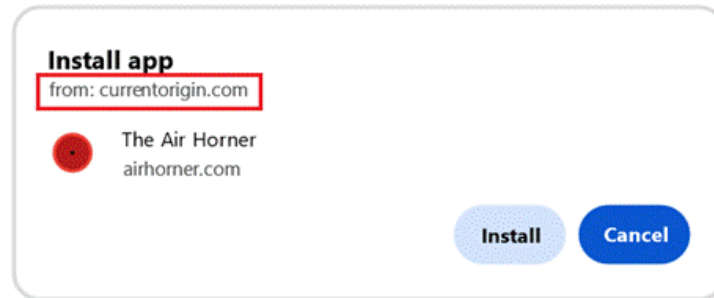
(Now) User flow

Note: assuming the site invoking the API is already installed.



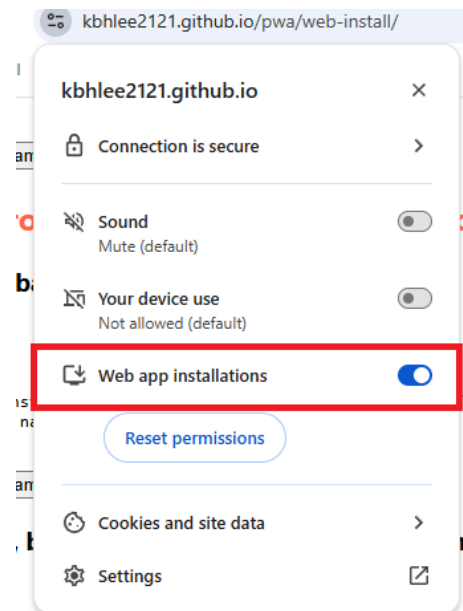
UX

1. Installation prompt – continue with current UX shown below.

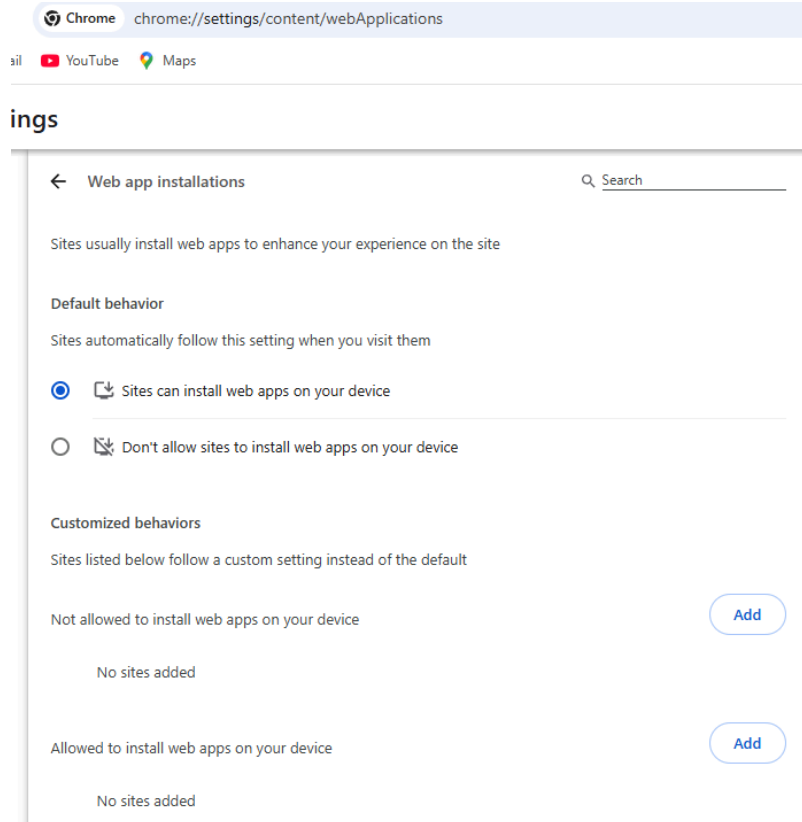


Note: A permission prompt will never be shown, but users still have the option to update the web install permission status from Allow (default) to Block in the Page Info Bubble (A) or site permission settings (B).

(A)



(B) Site permission settings



(During OT) Evaluate telemetry for signs of abuse, adjust API as needed

1. (Existing) UMA [WebApp.WebInstallApi.Result](#) (for web wide trend tracking)
  - a. (In progress) 2 UKMs (for individual bad actor tracking) - One UKM for the installer origin and the 2nd UKM for the installee origin.
2. (In progress) [installed\\_by](#) in [web\\_app.h - Chromium Code Search](#) - Records the installer origin to the installed app's web app database.
3. (Existing) UMA [Webapp.Install.InstallBounce](#)
4. (Existing) [webapps.InstallResultCode](#)

(Later) Evaluate shipping a 2nd, declarative, PEPC version of PWA install

*Note: This is not OT blocking for the current imperative version but something we are evaluating while working towards OT.*

Wait for more widespread adoption and standardization of PEPC. Treat supporting PEPC as a separate feature.

1. (In progress) Write an explainer for the PEPC version of Web Install to continue PEPC discussions.

## After OT starts:

1. Work with Serena and Erin on UX improvements for the install prompt and potentially other UX areas. UX updates are expected to be handled after OT starts.

## Considerations

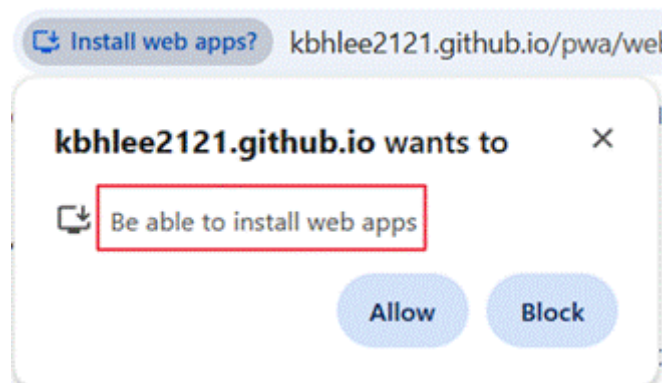
---

### Previously proposed

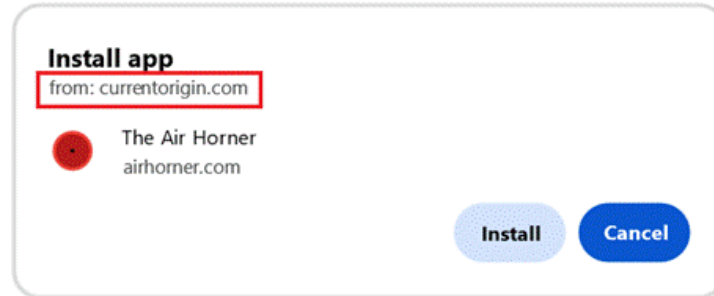
Date: Sep 8, 2025

UX

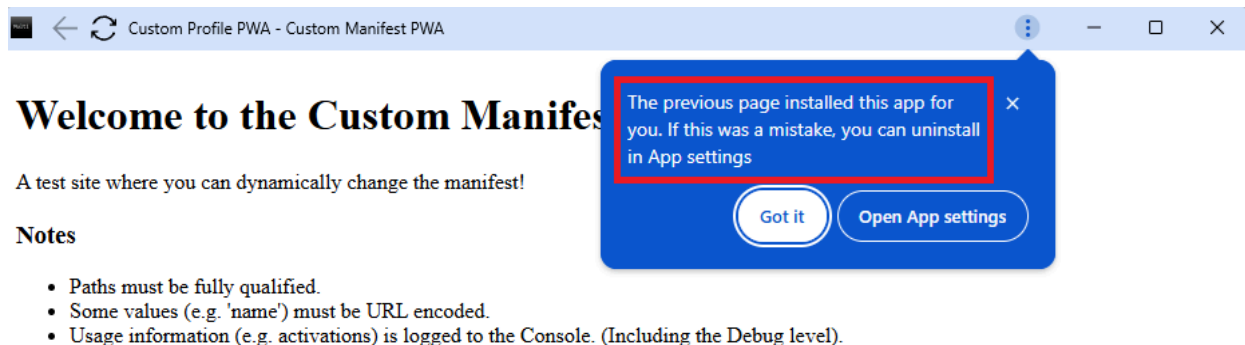
1. Permission prompt – change to **"Install other sites' web apps"**



2. Installation prompt – change to **"the current page wants to install"**



3. IPH bubble – "[origin] installed this app for you. If this was a mistake, you can uninstall in App settings"





Tab 6

