FIREFOX OS AND APPS SECURITY

Lucas Adamski <lucas@mozilla.com>

SECURITY PRINCIPLES

- Protect the OS from malicious apps
- Protect apps from each other
- Only share the user's private data with consent

https://wiki.mozilla.org/Apps/Security

mozilla

FIREFOX OS == B2G

Firefox OS == FFOS == B2G

FIREFOX OS STACK

- Apps
- Gaia – UI
- Gecko – Browser Runtime
- Gonk – Underlying Linux OS

GAIA

- User interface for Firefox OS
- Apps: System, Homescreen & core apps required for phone usage (dialer, SMS, email, camera, music, etc.)
- Entirely HTML, CSS, JavaScript, only interface to underlying operating system is through Gecko Web APIs
- Third-party Apps can be installed alongside Gaia

GECKO

- The "application runtime” for Firefox OS (as well as Firefox, Fennec and Thunderbird)
- Gecko implements the open standards for HTML, CSS, and JS and makes those interfaces run well on all the OSes that Gecko supports
- Gecko consists of, among other things, a networking stack, graphics stack, layout engine, virtual machine (for JS), and porting layers

GONK

- The lower-level "operating system" of FFOS
- Linux kernel and userspace hardware abstraction layer (HAL).
- Common open-source kernel and several userspace libraries: linux, libusb, bluez, etc.
- Parts of the HAL are shared with the android project: GPS, camera, among others.
- Gonk is a **porting target** of Gecko (in the same way as there are Gecko ports for Mac OS X or Android)

mozilla

BUNDLED & 3RD PARTY APPS

- All in the form of Open Web Apps
- HTML, JavaScript, CSS

mozilla

FIREFOX OS AND APPS

mozilla

TYPES OF APPS IN THE WORLD

- Websites
- Websites bookmarked on your phone homescreen
- Installed apps on your phone
- Installed apps on your computer

NOT JUST FOR FIREFOX OS

- FirefoxOS
- WebRT (Android)
- Firefox (Desktop)

TYPES OF APPS

- Web Content
- Installed Web Apps
- Privileged Apps
- Certified Apps

  All just HTML, CSS, JS

REGULAR WEB CONTENT

- Normal web sites should have access to as many rich APIs as can be safely exposed to arbitrary content
- Only expose APIs that are safe to be potentially called by any website (with user consent)
- Permissions not remembered by default

mozilla

INSTALLED WEB APPS

Essentially the same security model and behavior as normal web content, installable from anywhere

- Website plus a manifest file
- Manifest must be "application/x-web-app-manifest+json"
- Direct fullscreen access
- Higher storage quotas
- Can request access to: Geolocation API, Sensor API, Alarm API, FM Radio

PRIVILEGED APPS

Equivalent in security and functionality to native apps on other mobile platforms

- ZIP file format with manifest
- app://
- Content Security Policy
  - default-src *; script-src 'self'; object-src 'none'; style-src 'self' 'unsafe-inline'
- Reviewed & signed by trusted app store
- Ability to directly access higher-risk APIs: Alarm API, TCP Socket, Contacts API, Device Storage API, Browser API, WiFi Information API

CERTIFIED APPS

Intended for system-critical applications (i.e. Gaia).
Very similar to privileged model except

- Ability to access system-critical APIs
- Content Security Policy
  - default-src *; script-src 'self'; object-src 'none'; style-src 'self'
- Never prompt for access (implicit)
- Direct access to: Background services, WebSMS, WebTelephony, WebBluetooth, MobileConnection API, Power Management API, Push Notifications API, Settings API, Permissions API

mozilla

MANAGING APP DATA

Per-app separation of

- cookies
- localStorage
- appCache
- indexDB

LAUNCHING APPS

Applications can be started in following ways

- Manually by the user (tap homescreen icon and system app manager launches the app)
- Background apps launched at startup by System App
- Web Activities, Alarm API, Notification API
- Embedding content does NOT run the app
- Apps cannot launch other apps directly

CLOSING APPS

Applications can be closed in following ways:

- Manually closed (via task manager)
- System will kill background apps when memory is low

<IFRAME MOZAPP>

Apps run inside mozApp iframes:

- Only the system app may embed these type of iframes
- Creates a separate data jar for the App (separate principal)
- Apps run inside a content process (not parent process)

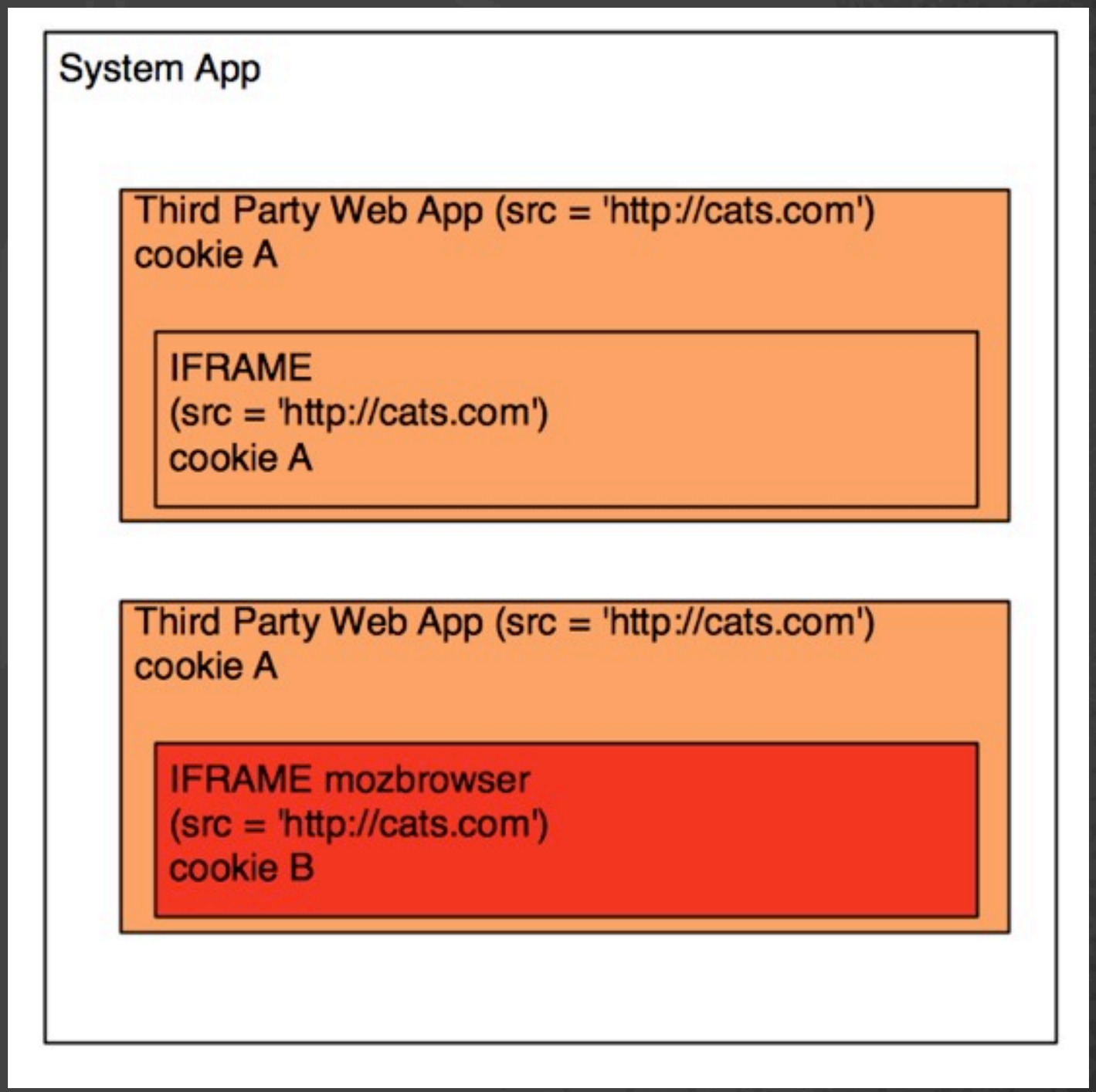
<IFRAME MOZBROWSER>

For apps which will load a lot of other web pages

- In child, window.top==window (!window.parent)
- Parent frame allowed limited cross-domain access, such as getting current location, listen for certain child events

TYING IT TOGETHER

- System embeds app
- App embeds iframes
- Separate cookies for iframe vs. mozBrowser



PERMISSIONS

OBTAINING PERMISSION

All apps follow the same permission model

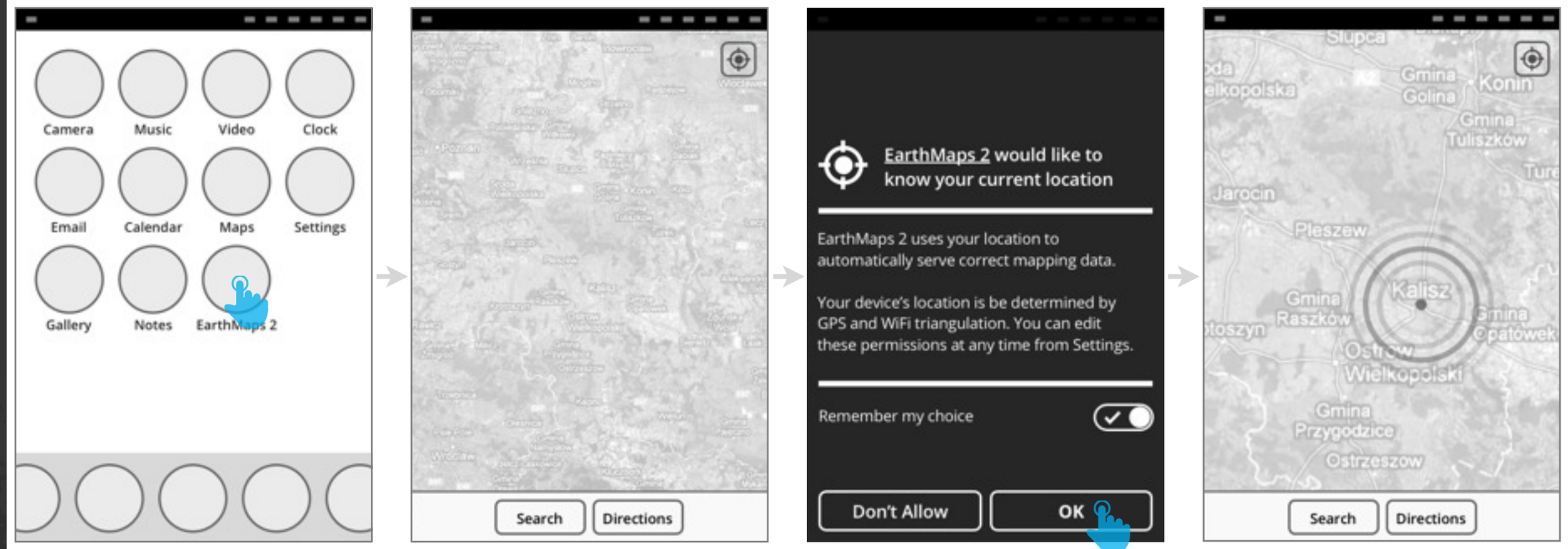
- Permissions need to be declared in manifest
- Implicit permissions: granted without prompt
- Explicit permissions: granted by user consent

mozilla

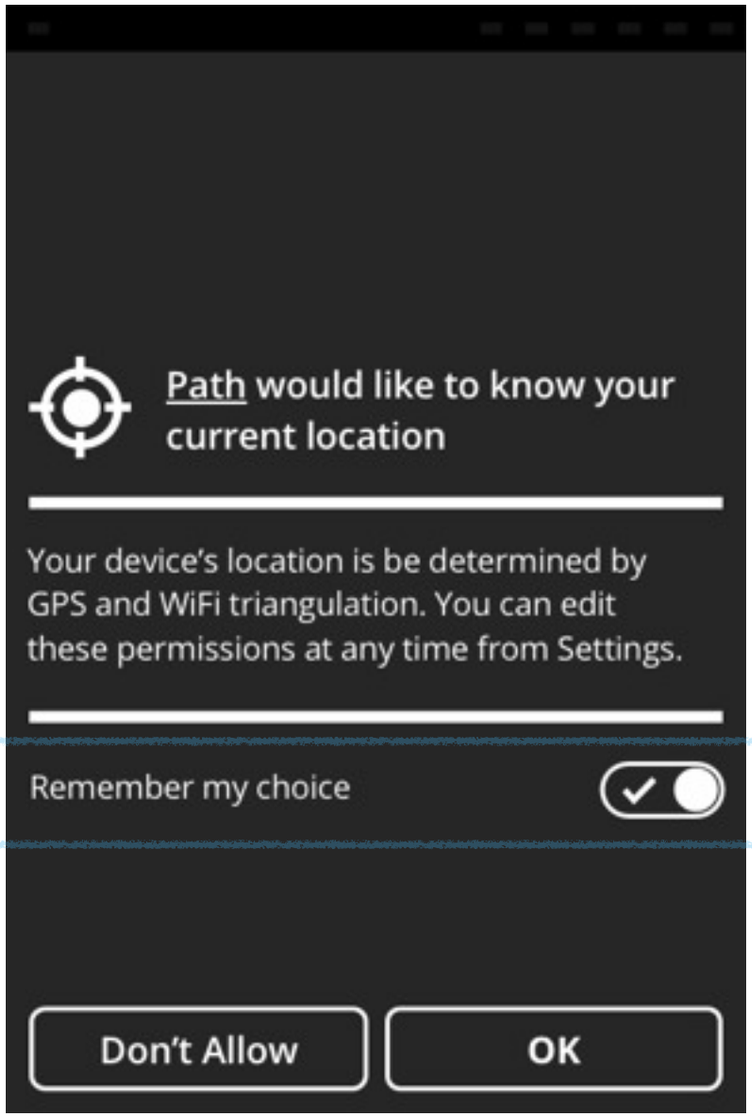
AT TIME OF USE

Prompt for permissions only when they're needed

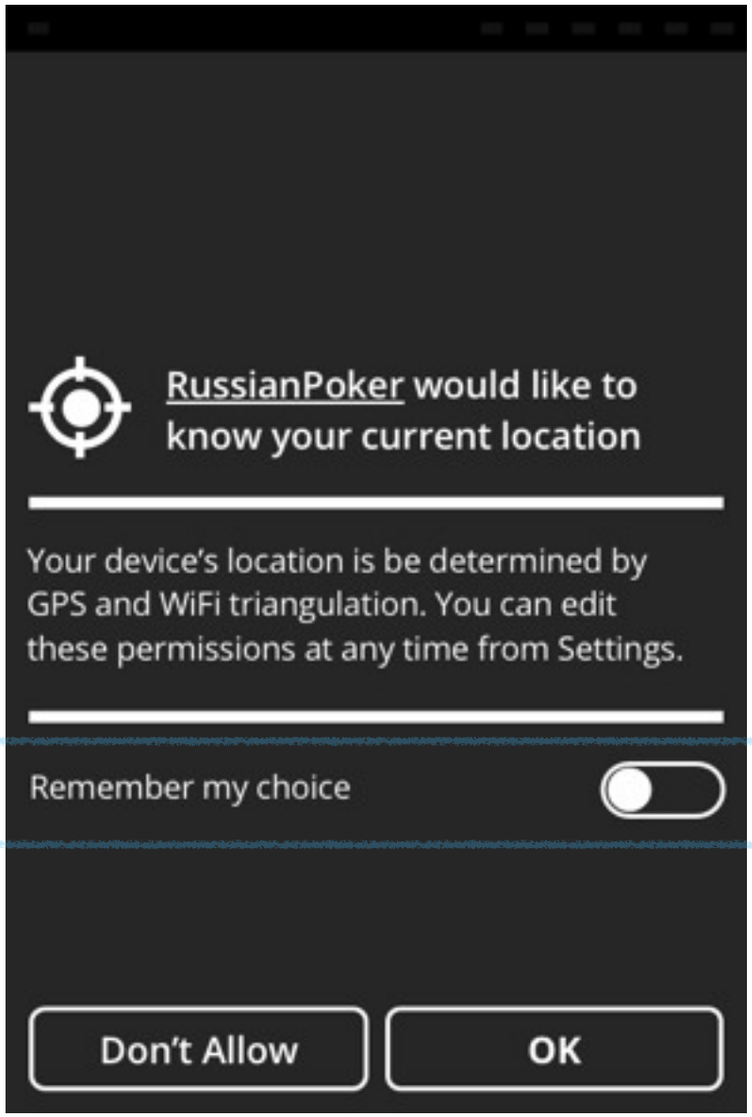
Permission prompts are managed automatically; app just calls a webAPI



REMEMBERING DECISIONS



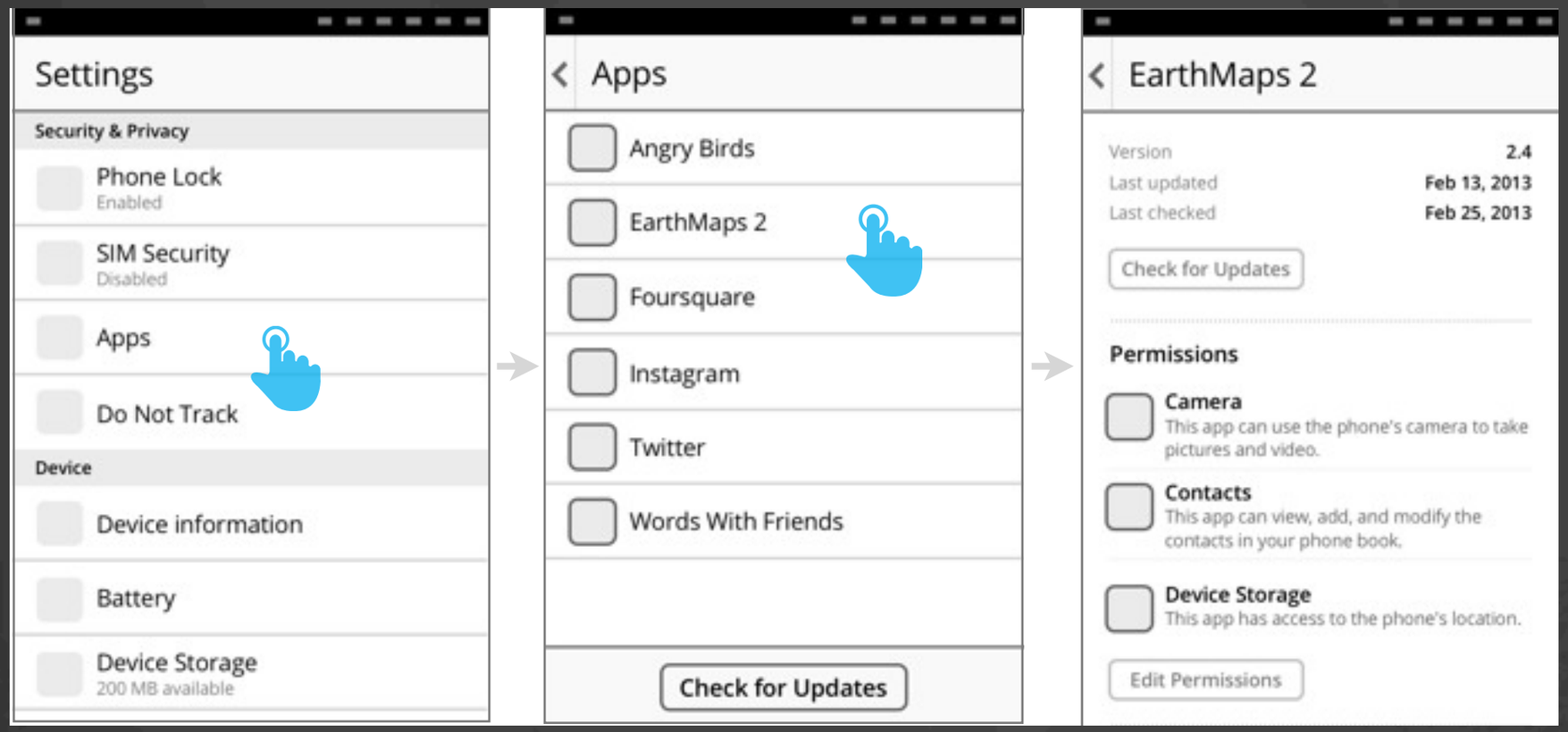
Priviledged app: "Remember my choice" is on by default.



Non-priviledged app: "Remember my choice" is off by default.

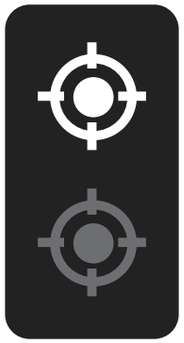
CHANGING YOUR MIND

Any decisions you make around permissions can be changed later via settings



NOTIFICATIONS

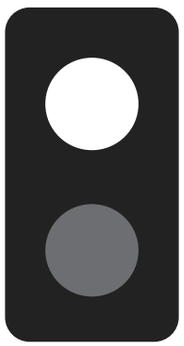
Geolocation



Active: an App (foreground or background) is currently using the Geolocation permission.

Recently-active: an App (foreground or background) has used the Geolocation permission in the last X minutes (time TBD).

Audio/Video Recording



Active: an App (foreground or background) is currently using the Audio/Video Recording permission.

Recently-active: an App (foreground or background) has used the Audio/Video Recording permission in the last X minutes (time TBD).

WEB ACTIVITIES

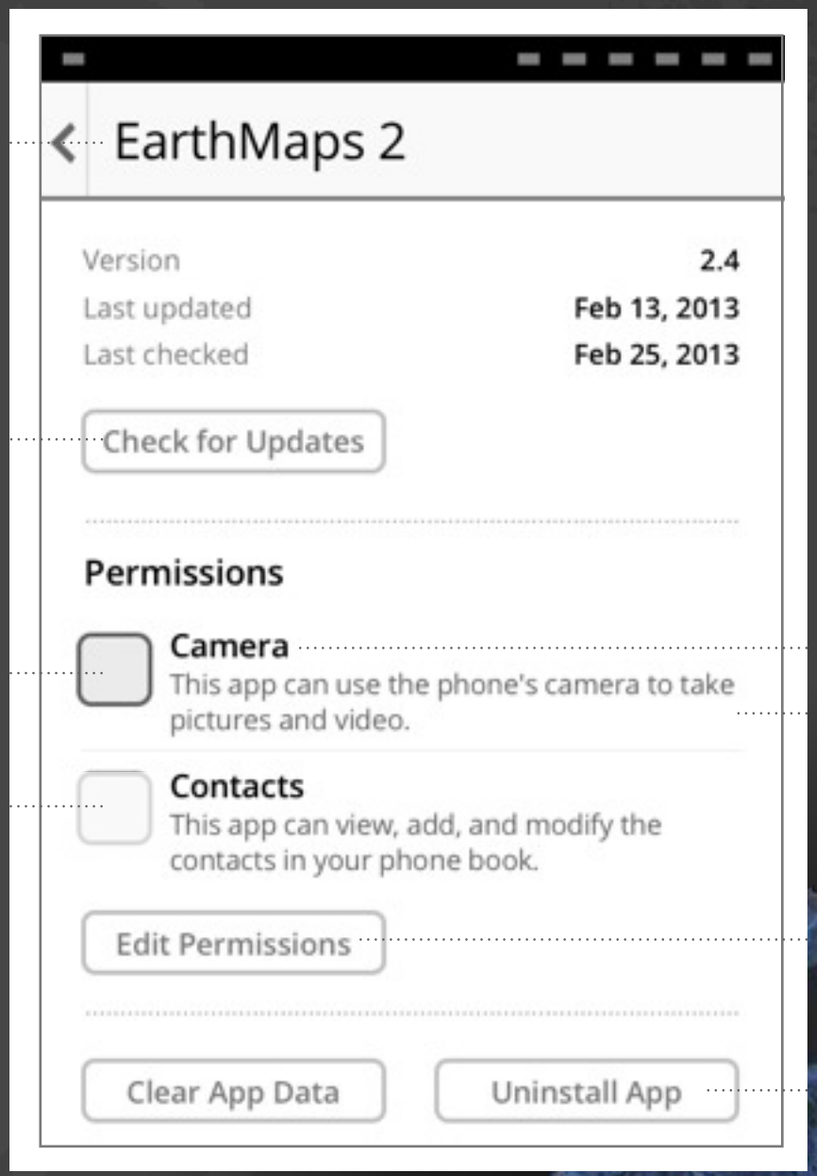
Let apps communicate with each other. Also useful when an app doesn't have direct access to an API.

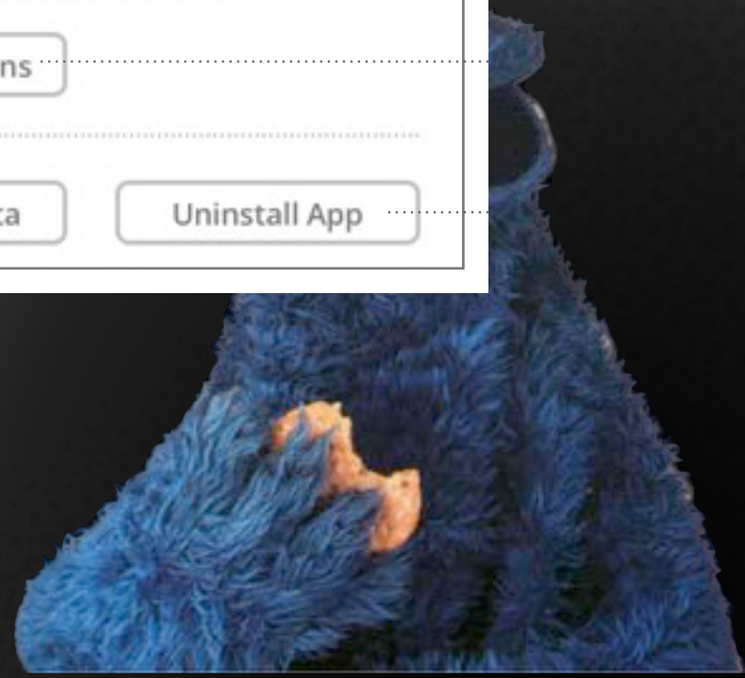
- Apps register to handle certain activities
- Other apps initiate an activity
- ex. Give me a picture
- ex. Give me a contact
- ex. Send an SMS

mozilla

PRIVACY

- Do-Not-Track
- Ask user for access
- Manage and clear data per-app





PROTECTING THE OS

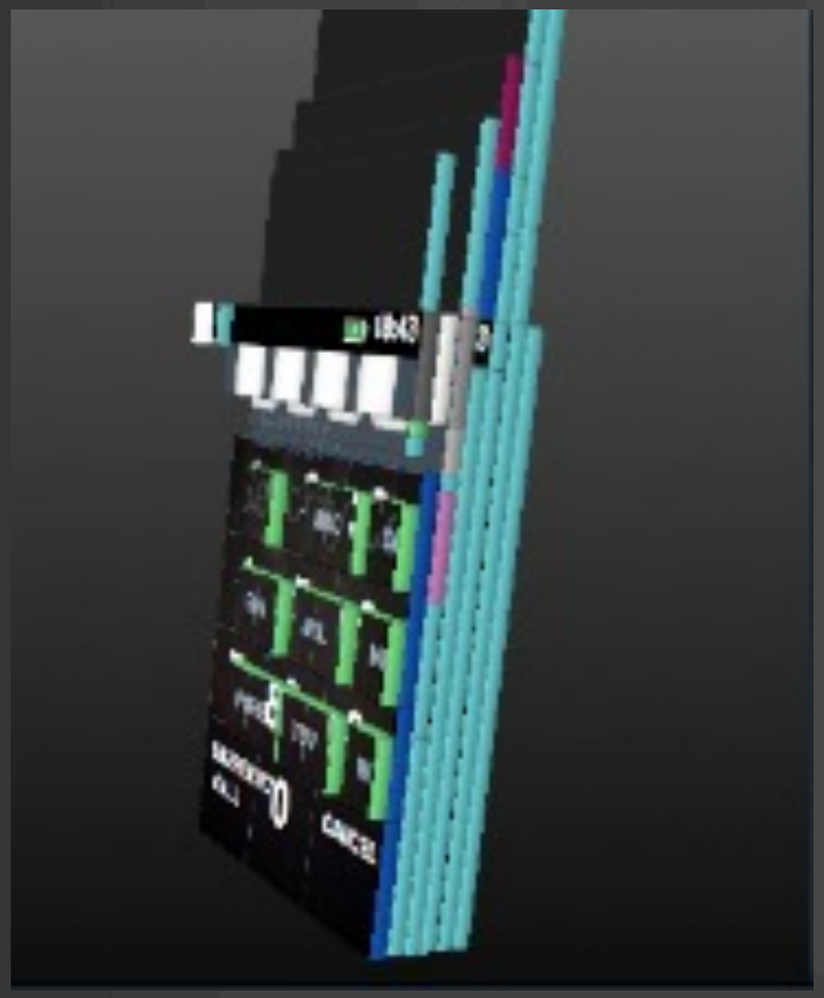
BOOT SEQUENCE

1. Linux startup
2. FFOS (Gecko)
3. Shell.xul
4. Gaia System App
5. Homescreen App
6. Installed Apps Register

STRUCTURE

<window> Gecko chrome (shell.xul)
    <iframe> system app
        <iframe> homescreen app
        <iframe> keyboard
        <iframe> lockscreen
        ... more app iframes are created here as apps are loaded

DIALER AND SYSTEM APPS



mozilla

PROTECTING APPS FROM APPS

Each app runs in its own content process

- Apps cannot see each others data or cookies
- One app can’t directly launch or frame another
- App process permissions == manifest permissions

PROTECTING OS FROM APPS

The OS is protected from direct access by apps

- Apps can only talk to OS via IPC (IPDL)
- App content processes run as low-rights
- App process permissions are limited to itself

OS UPDATES

OS updates are separate from Gecko updates

- Underlying OS can be updated via OTA or USB (i.e. flashing)
- Gecko + Gaia updated via normal Mozilla release
- App updates are handled separately and individually

HOW TO GET INVOLVED

- irc.mozilla.org: #security #privacy
- usenet: mozilla.dev.security mozilla.dev.privacy
- https://wiki.mozilla.org/Apps/Security
- https://wiki.mozilla.org/Gaia/System/Updates
- https://wiki.mozilla.org/B2G/Architecture/Runtime_Security
- https://bugzilla.mozilla.org/show_bug.cgi?id=764189