Security, Privacy, & User Expectations:

Case Studies in Web Tracking and Application Permissions

Franziska Roesner

Assistant Professor
Computer Science & Engineering
University of Washington

UNIVERSITY of WASHINGTON

SECURITY AND PRIVACY RESEARCH LAB

University of Washington Computer Science & Engineering

Security, Privacy, & User Expectations:

Case Studies in Web Tracking and Application Permissions

Franziska Roesner + many collaborators!

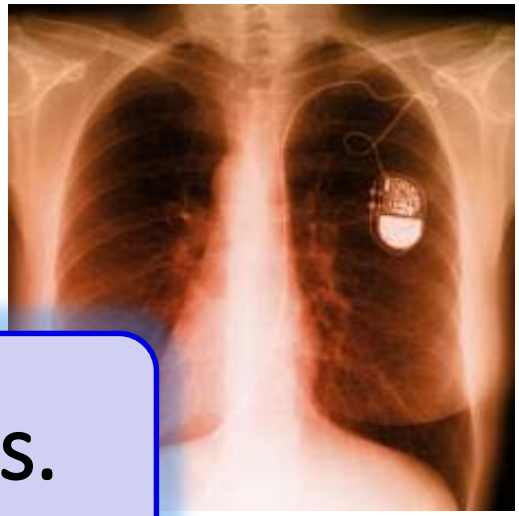
Assistant Professor
Computer Science & Engineering
University of Washington

UNIVERSITY of WASHINGTON

SECURITY AND PRIVACY RESEARCH LAB

University of Washington Computer Science & Engineering

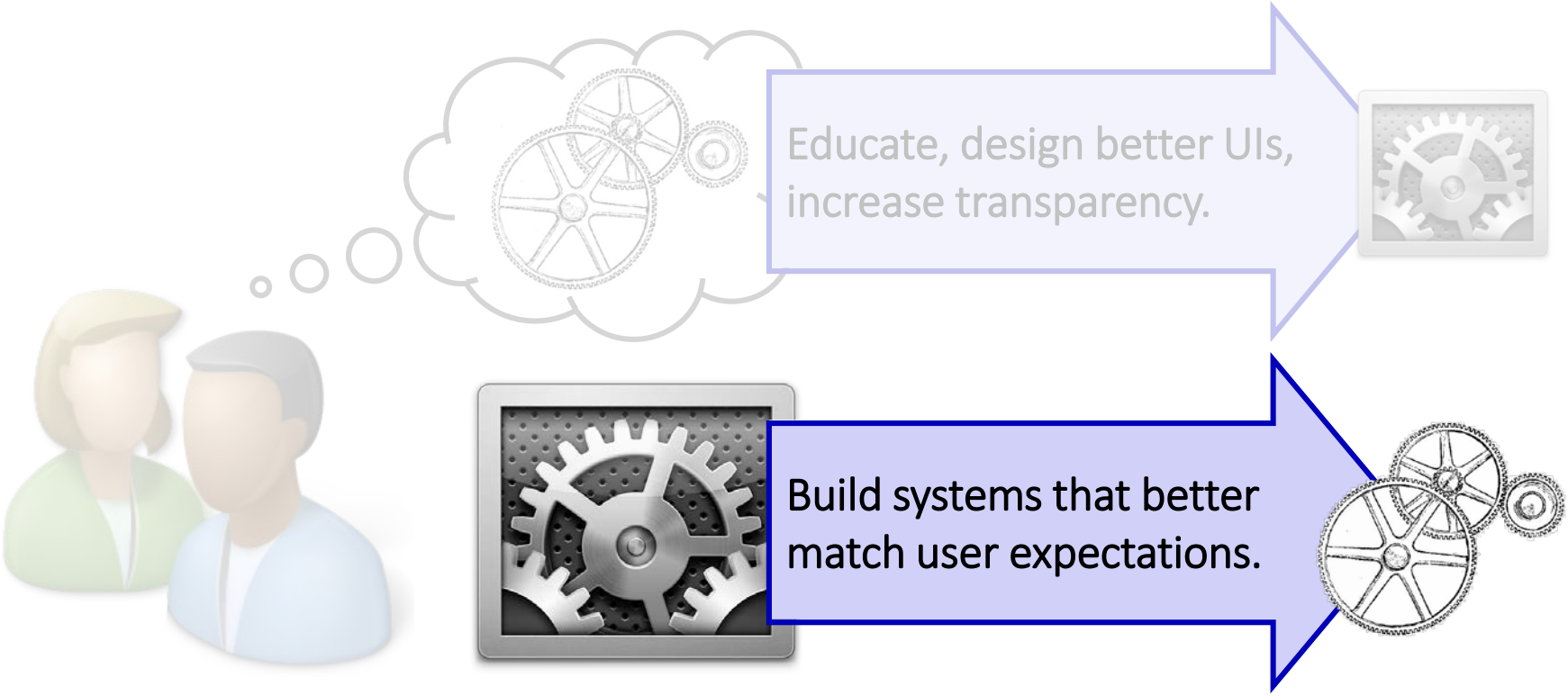
New technologies bring new benefits…



… but also new risks.

Improving Security & Privacy

Security and privacy challenges often arise when user expectations don't match real system properties.



Outline

I. **The Web:**
   Third-Party Tracking

II. **Modern OSes:**
   Permission Granting

Outline

I. The Web:
Third-Party Tracking

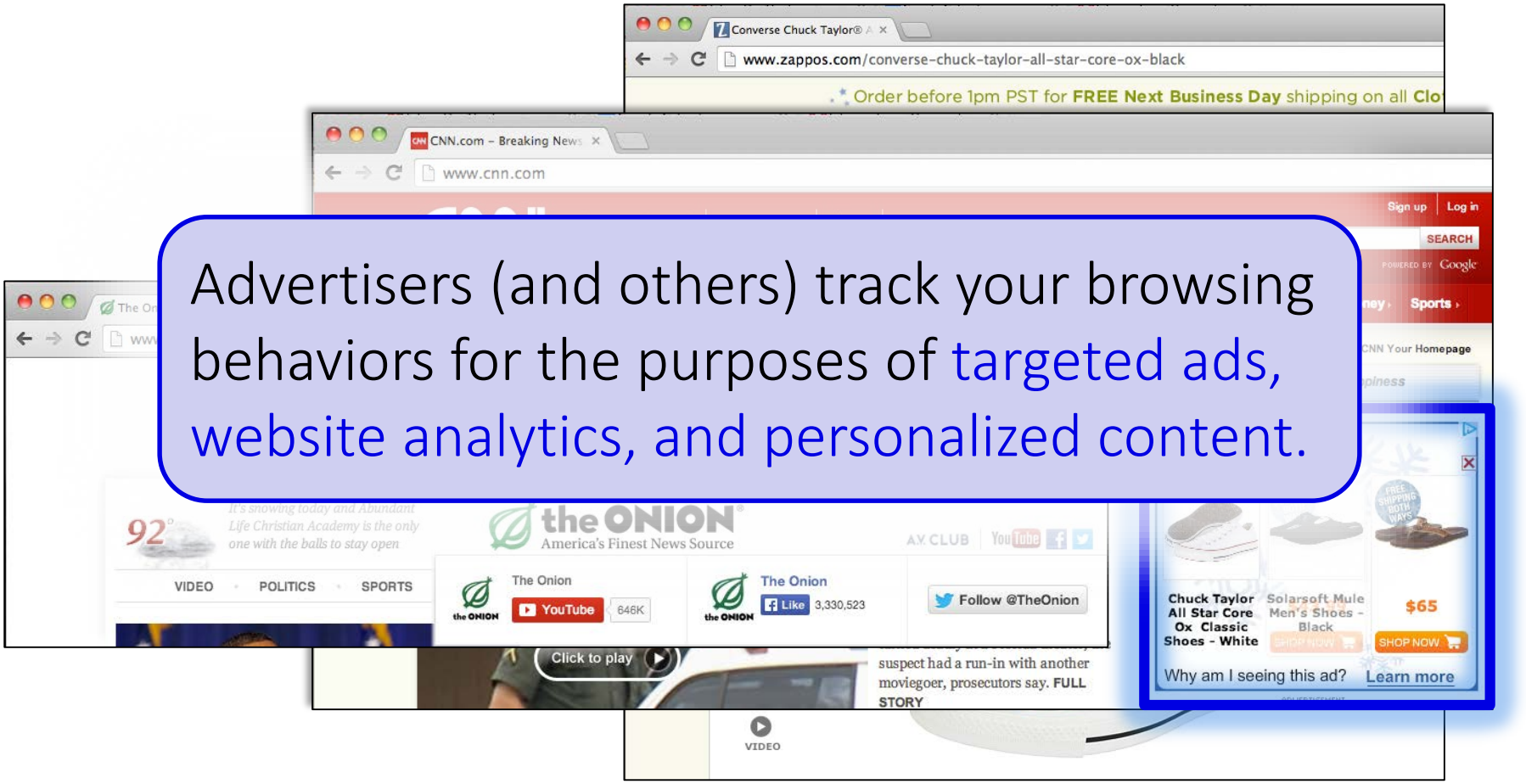
II. Modern OSes:
Permission Granting

F. Roesner, T. Kohno, D. Wetherall. "Detecting and Defending Against Third-Party Tracking on the Web." In USENIX Symposium on Networked Systems Design and Implementation (NSDI) 2012.

F. Roesner, C. Rovillos, T. Kohno, D. Wetherall. "ShareMeNot: Balancing Privacy and Functionality of Third-Party Social Widgets." In USENIX ;login: 2012.

A. Lerner, A. Kornfeld Simpson, T. Kohno, and **F. Roesner**. "Internet Jones and the Raiders of the Lost Trackers: An Archaeological Study of Web Tracking from 1996 to 201." In USENIX Security Symposium 2016.

Ads That Follow You



Advertisers (and others) track your browsing behaviors for the purposes of targeted ads, website analytics, and personalized content.

Third-Party Web Tracking

These ads allow **criteo.com** to link your visits between sites, even if you never click on the ads.

Concerns About Privacy

THE WALL STREET JOURNAL.

WHAT THEY KNOW | JULY 30, 2010

The Web's New Gold Mine: Your Secrets

A Journal inv

bus

CNN

Your Privacy

Big

dep

By

Hid

all to be put up

The file consists

identifies her as

Log In

The New York Times

May 6, 2011, 5:01 pm | 3 Comments

'Do Not Track' Privacy Bill Appears in Congress

By TANZINA VEGA

And the privacy legislation just keeps on coming.

On Friday, two bills were introduced in Washington in support of a Do Not Track mechanism that would give users control over how much of their data was collected by advertisers and other online companies.

als

tion

By JENNIFER VALENTINO-DEVRIES,
JEREMY SINGER-VINE and ASHKAN SOLTANI
December 24, 2012

Understanding the Tracking Ecosystem

In 2011, much discussion about tracking, but limited understanding of how it actually works.

Our Goal: systematically study web tracking ecosystem to inform policy and defenses.

Challenges:

- No agreement on definition of tracking.
- No automated way to detect trackers. (State of the art: blacklists)

Our Approach

ANALYZE

(1) Reverse-engineer trackers' methods.
(2) Develop tracking taxonomy.

MEASURE

(3) Build automated detection tool.
(4) Measure prevalence in the wild.
(5) Evaluate existing defenses.

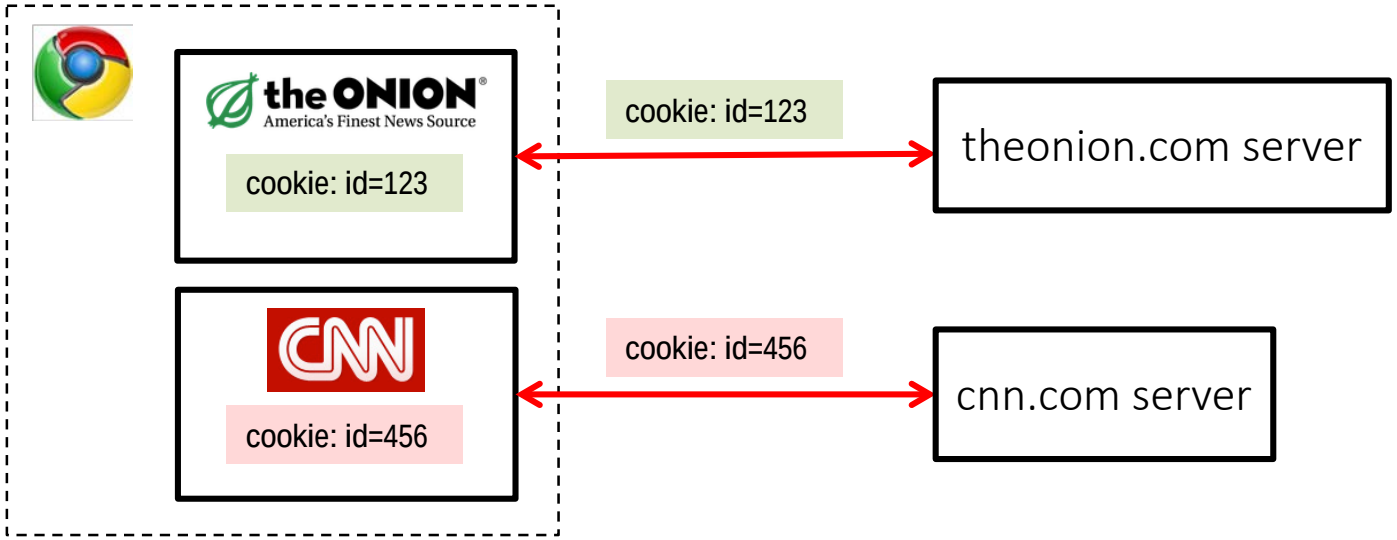
BUILD

(6) Develop new defenses.

Web Background

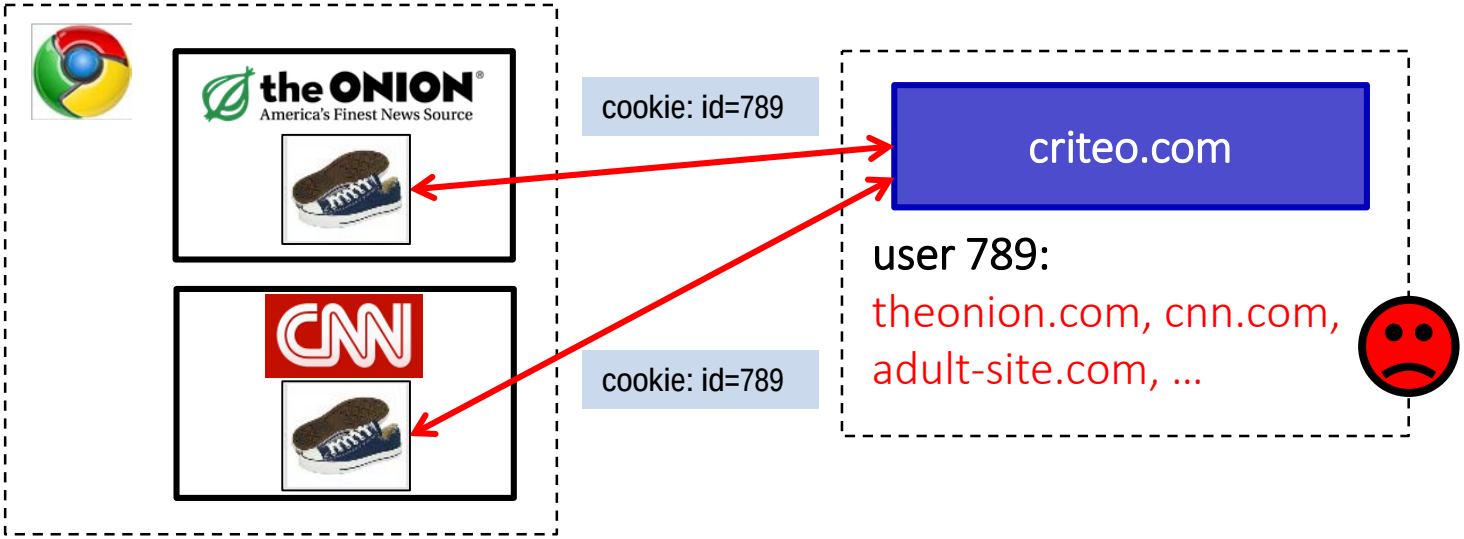
Websites store info in cookies in the browser.

- Only accessible to the site that set them.
- Automatically included with web requests.



Anonymous Tracking

Trackers included in other sites use cookies containing unique identifiers to create browsing profiles.



Our Tracking Taxonomy *[NSDI '12]*

In the wild, tracking is much more complicated.

(1) Trackers don't just use cookies.

- Flash cookies, HTML5 LocalStorage, etc.

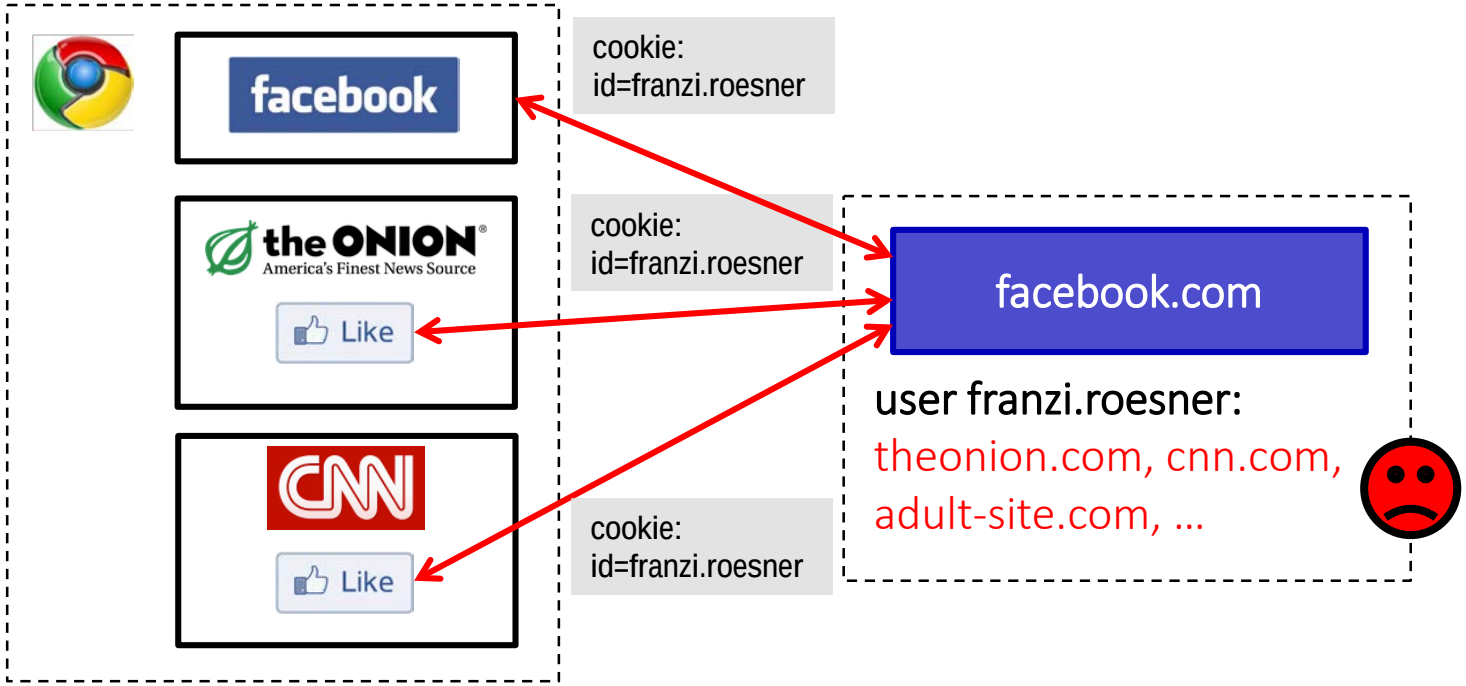
(2) Trackers exhibit different behaviors.

- Within-site vs. cross-site.
- Anonymous vs. non-anonymous.
- Specific behavior types:
  analytics, vanilla, forced, referred, personal.

Other Trackers?

"Personal" Trackers

Personal Tracking



- Tracking is not anonymous (linked to accounts).
- Users directly visit tracker's site → evades some defenses.

Measurement Study

Questions:

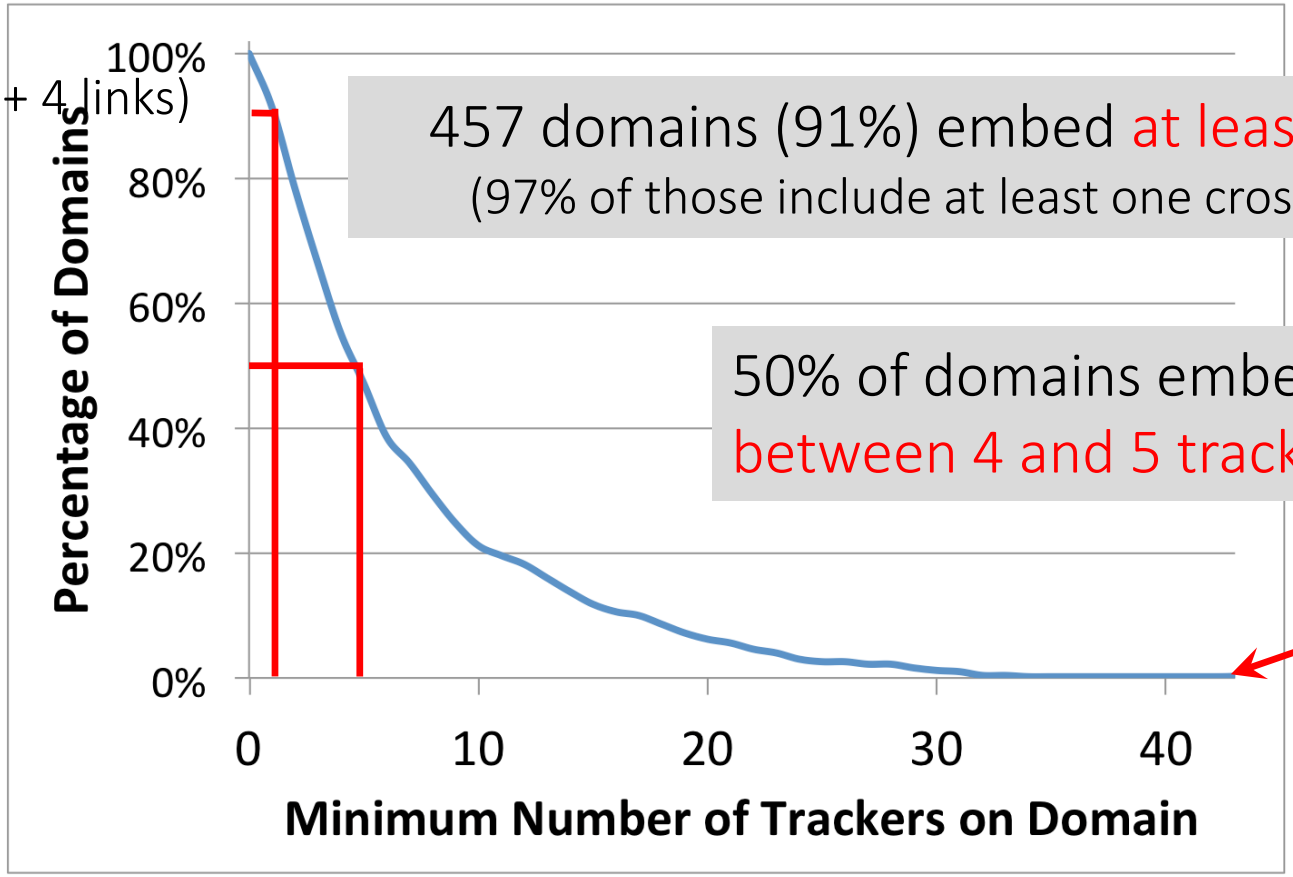
- How prevalent is tracking (of different types)?
- How much of a user's browsing history is captured?
- How effective are defenses?

Approach: Build tool to automatically crawl web, detect and categorize trackers based on our taxonomy.

TrackingObserver: **tracking detection platform**
http://trackingobserver.cs.washington.edu

How prevalent is tracking? (2011)

524 unique trackers on Alexa top 500 websites (homepages + 4 links)



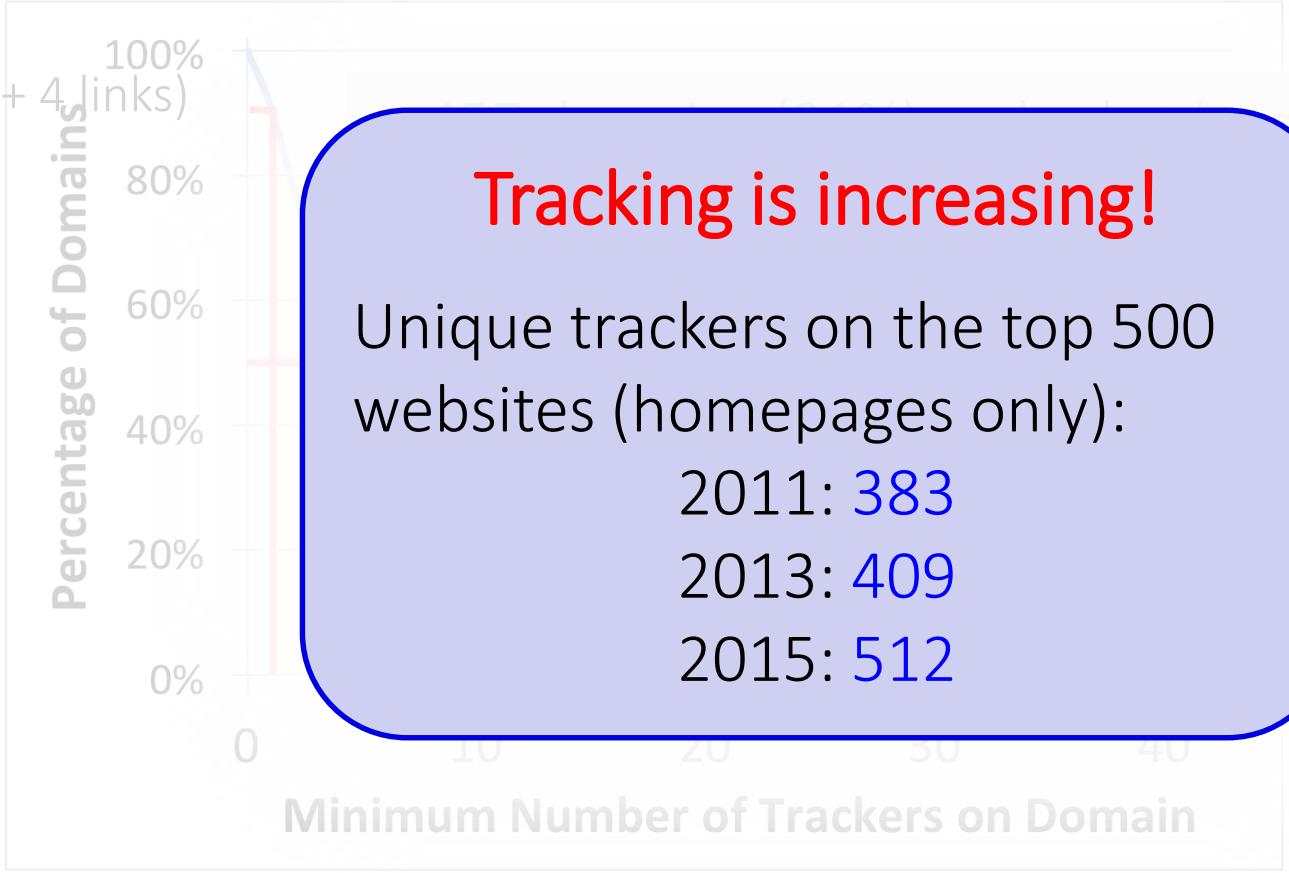
457 domains (91%) embed at least one tracker.
(97% of those include at least one cross-site tracker.)

50% of domains embed between 4 and 5 trackers.

One domain includes 43 trackers.

How prevalent is tracking? (2011)

524 unique trackers on Alexa top 500 websites (homepages



Tracking is increasing!

Unique trackers on the top 500 websites (homepages only):

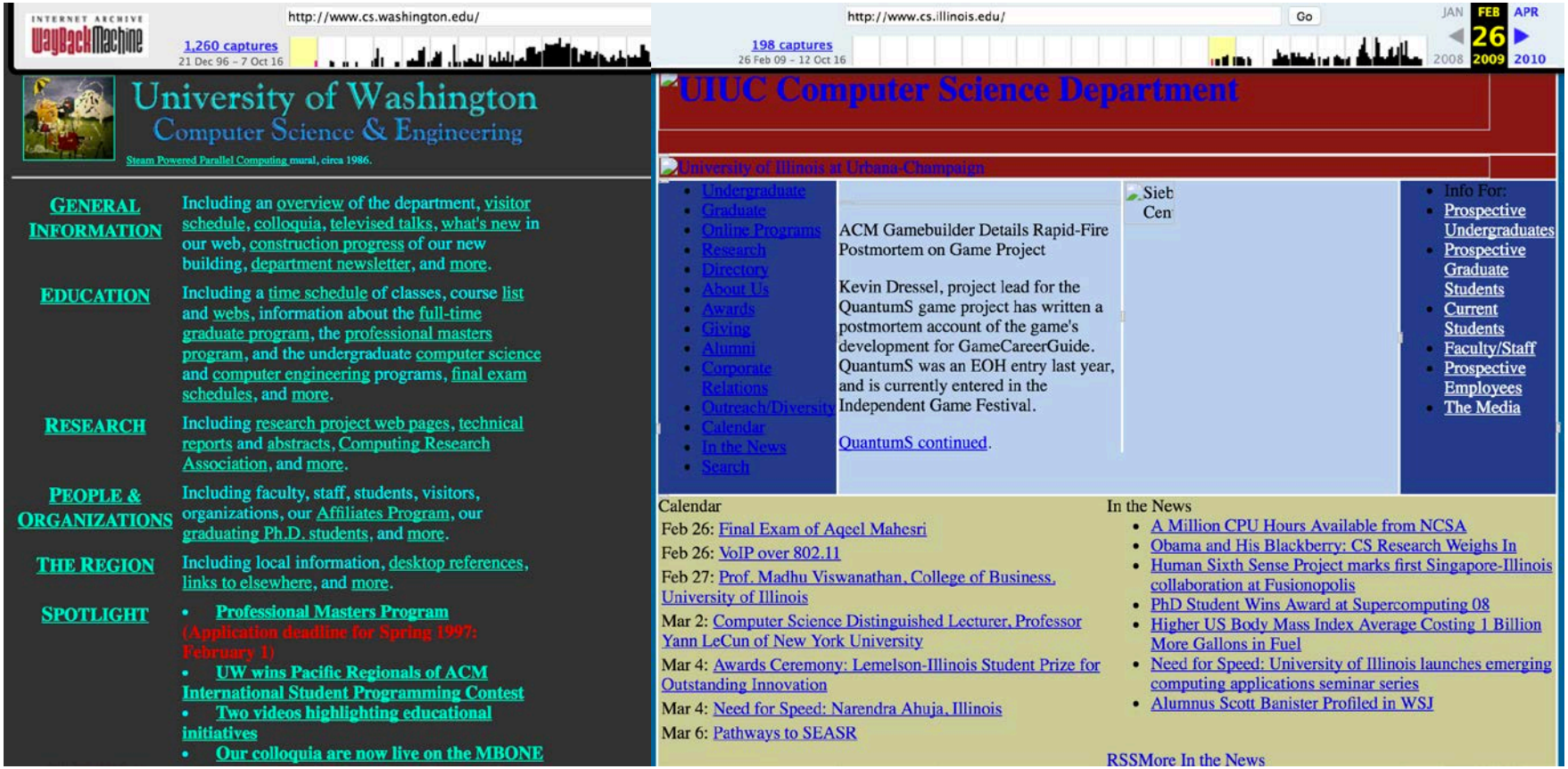
2011: 383
2013: 409
2015: 512

How has this changed over time?

- The web has existed for a while now…
  - What about tracking before 2011? (our first study)
  - What about tracking before 2009? (first academic study)

Solution: time travel!

The Wayback Machine to the Rescue

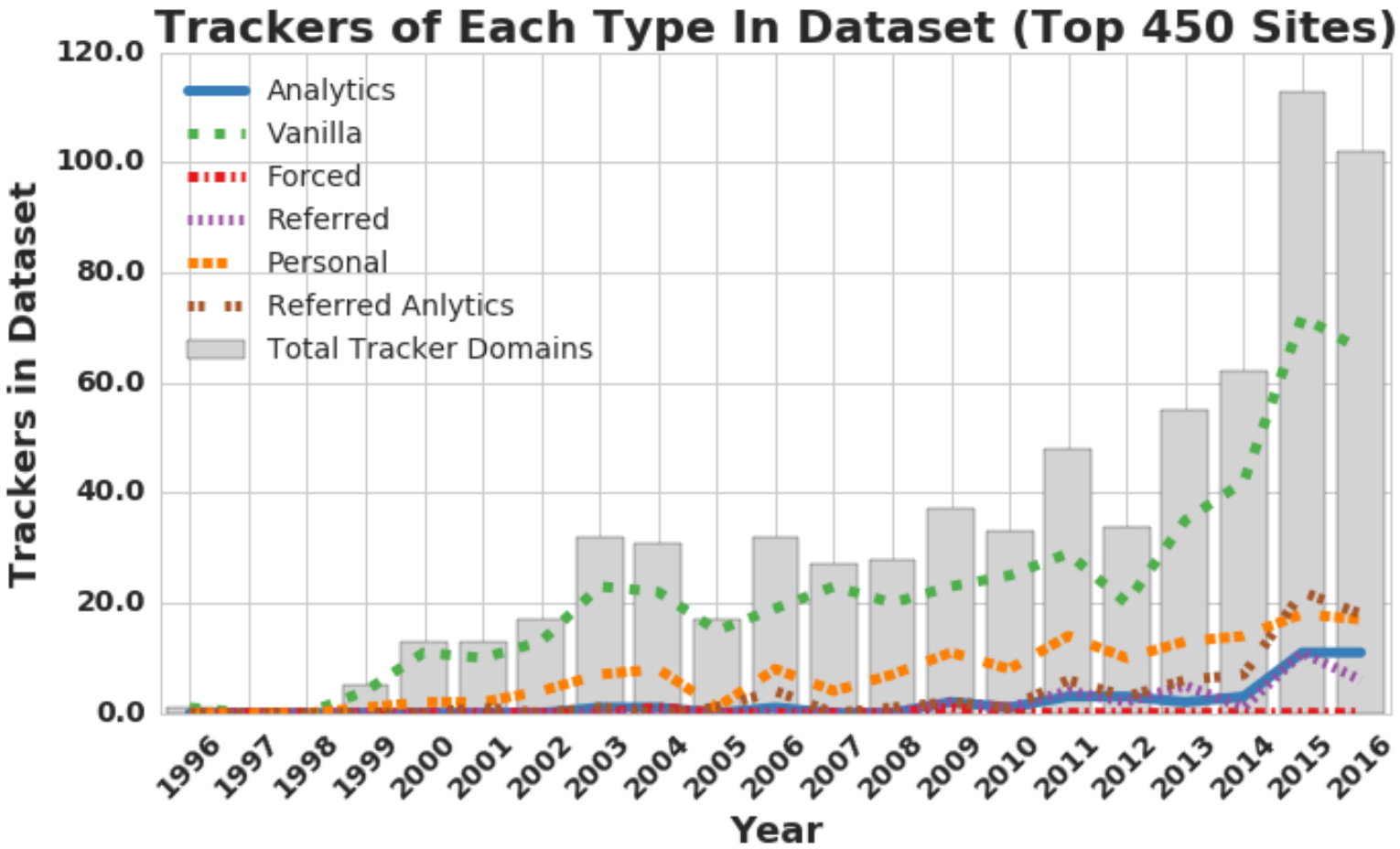


Time travel for web tracking (lots of challenges!)

http://trackingexcavator.cs.washington.edu

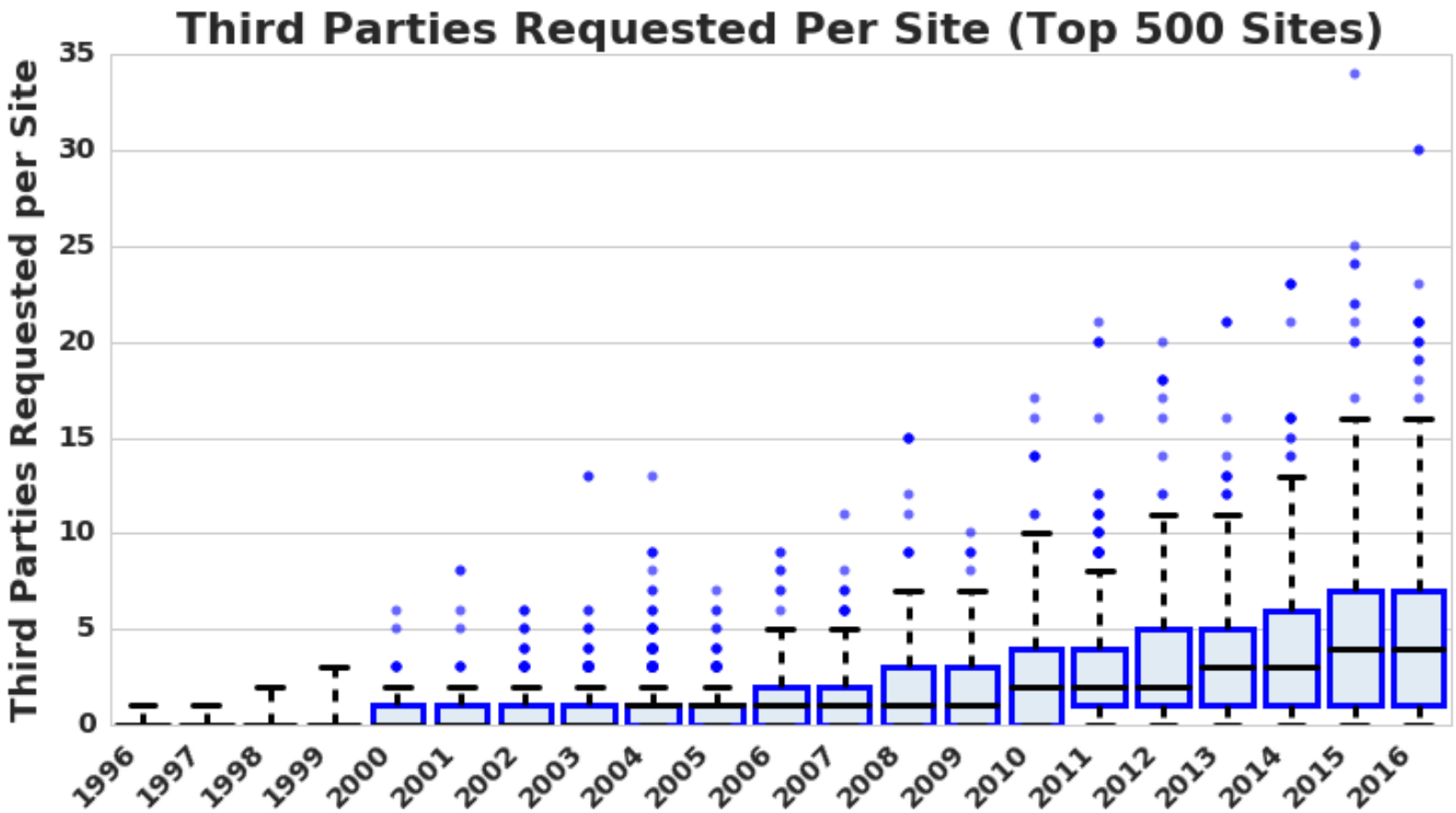
1996-2016: More & More Tracking

More trackers of more types



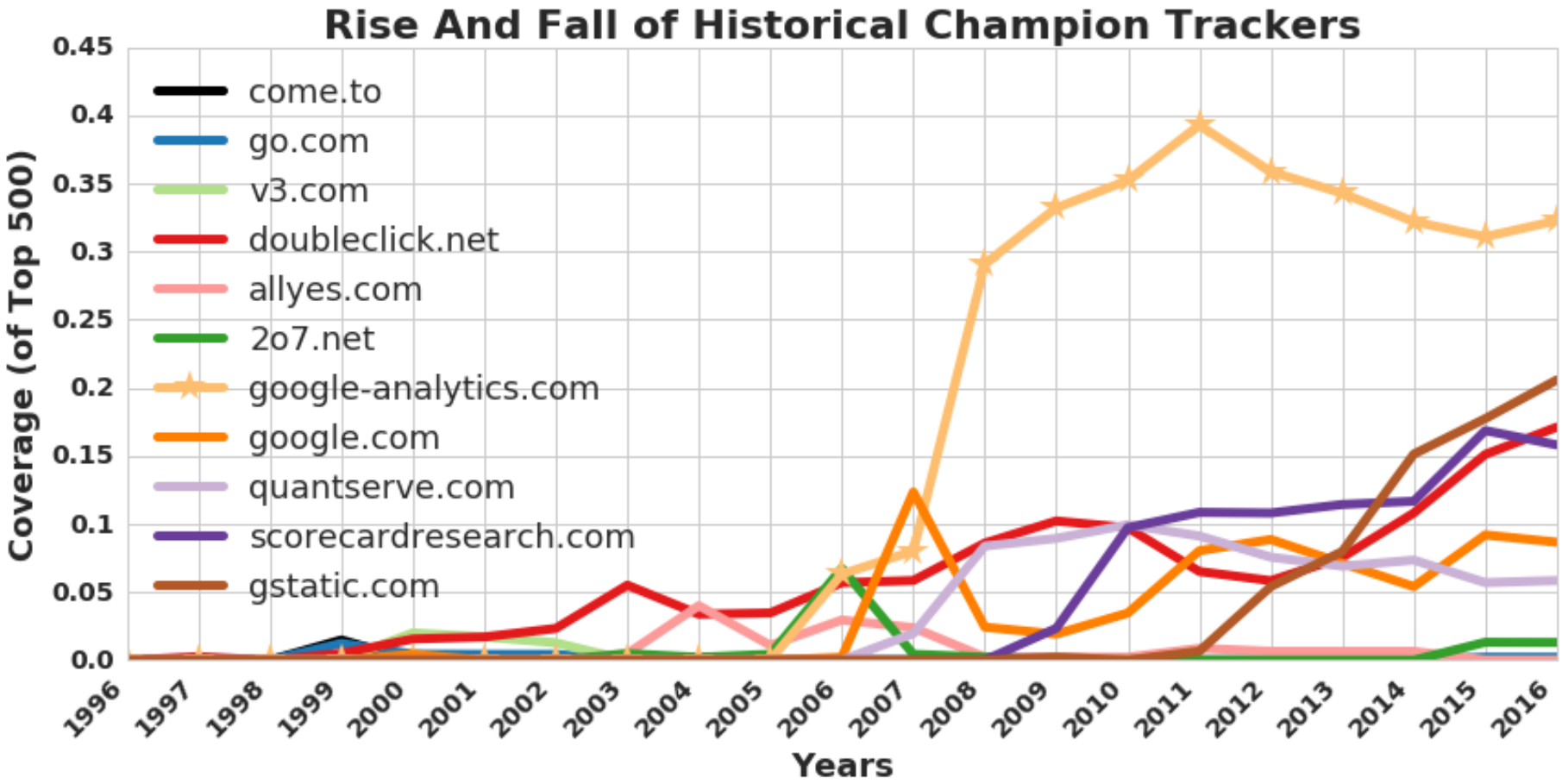
1996-2016: More & More Tracking

More trackers of more types, more per site

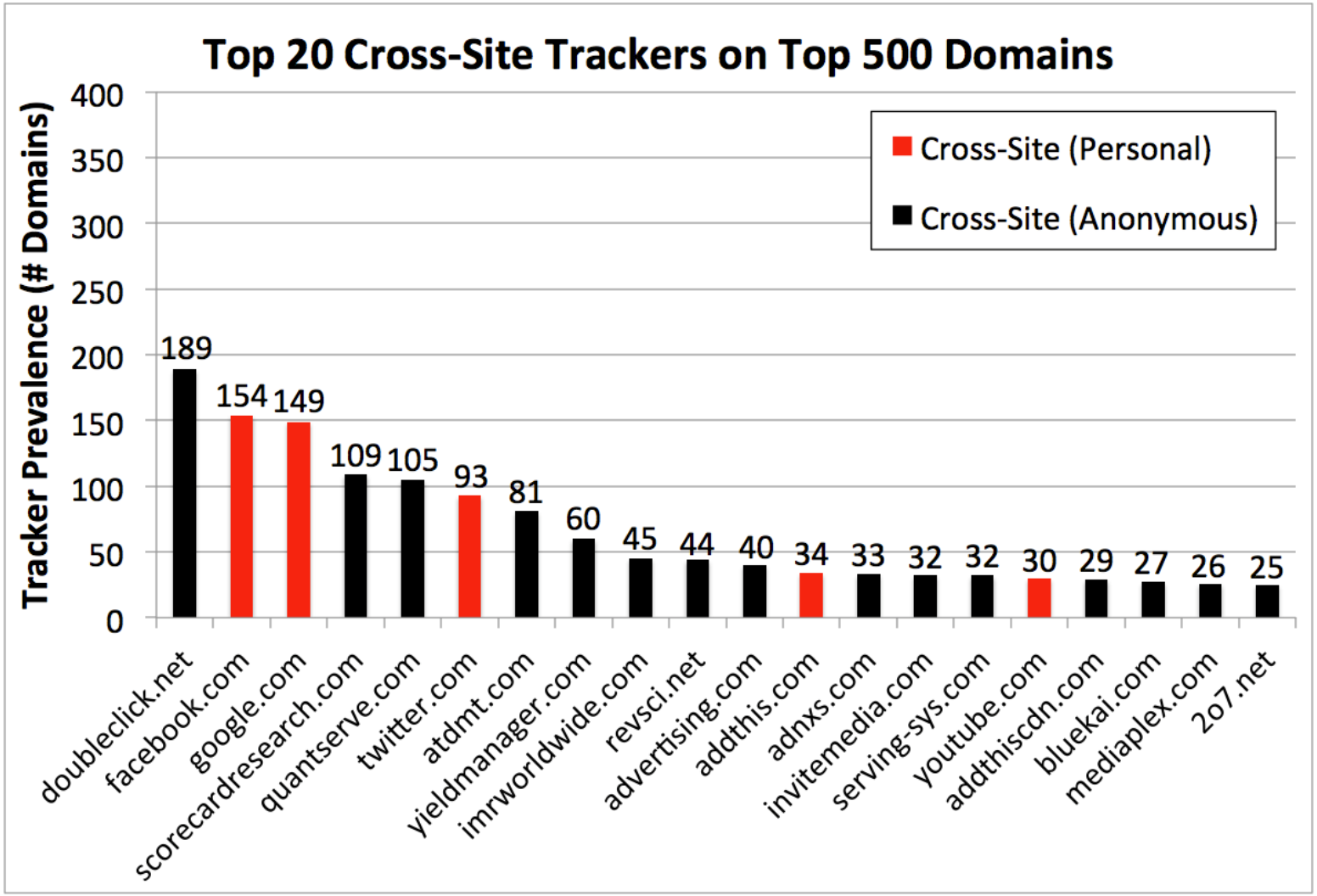


1996-2016: More & More Tracking

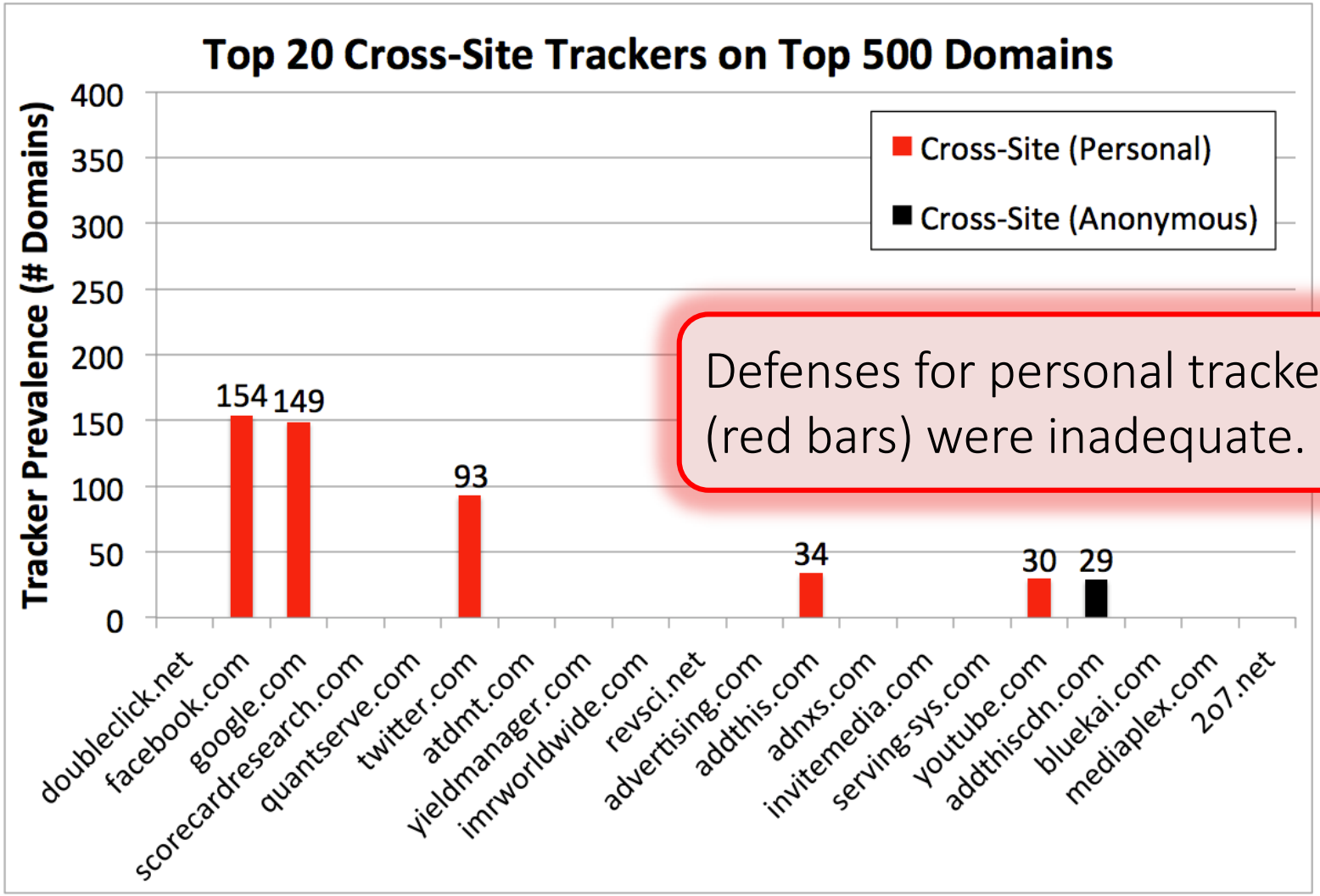
More trackers of more types, more per site, more coverage



Who/what are the top trackers? (2011)



Who/what are the top trackers? (2011)



Defenses for personal trackers (red bars) were inadequate.

Defense: ShareMeNot

Prior defenses for personal trackers: ineffective or completely removed social media buttons.

Our defense:

- ShareMeNot (for Chrome/Firefox) protects against tracking without compromising button functionality.
- Blocks requests to load buttons, replaces with local versions. On click, shares to social media as expected.
- Techniques adopted by Ghostery and the EFF.

http://sharemenot.cs.washington.edu

Summary: Web Tracking

Pre-2011: Limited understanding of web tracking.

Our work:

- Comprehensive tracking taxonomy.
- Measurements and archeological study from 1996-2016.
- Example results: >500 unique trackers, some able to capture up to 66% of a user's browsing history.
- New defense for "personal trackers" like Facebook, Google, Twitter: built into ShareMeNot, adopted by Ghostery + EFF.

Outline

I. The Web:
Third-Party Tracking

II. Moderns OSes:
Permission Granting

F. Roesner, T. Kohno, A. Moshchuk, B. Parno, H. J. Wang, C. Cowan. "User-Driven Access Control: Rethinking Permission Granting in Modern Operating Systems." In IEEE Symposium on Security & Privacy 2012 (Best Practical Paper Award).

F. Roesner, J. Fogarty, T. Kohno. "User Interface Toolkit Mechanisms for Securing Interface Elements." In ACM Symposium on User Interface Software and Technology (UIST) 2012.

F. Roesner, T. Kohno. "Securing Embedded User Interfaces: Android and Beyond." In USENIX Security 2013.

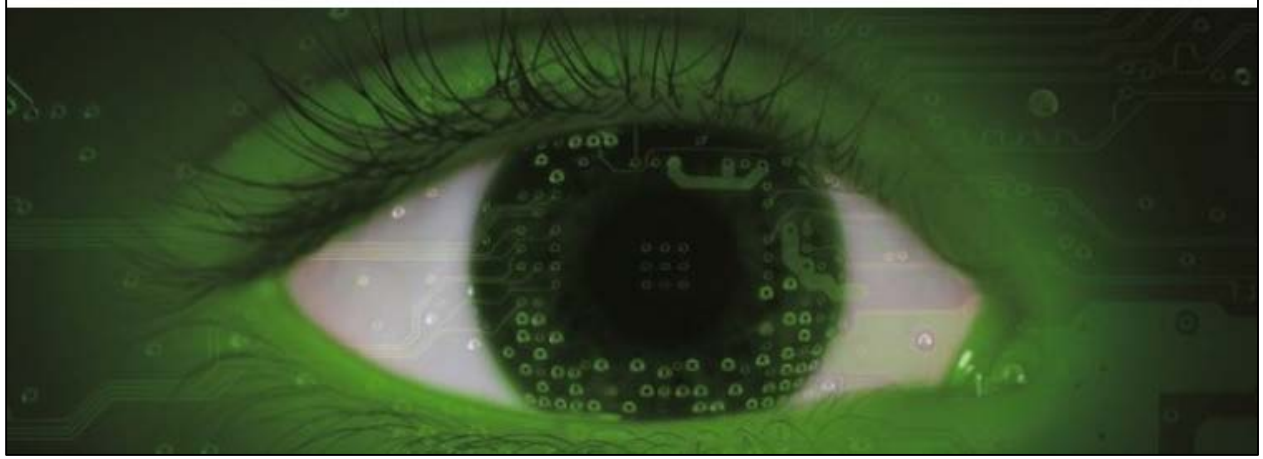
T. Ringer, D. Grossman, **F. Roesner**. "AUDACIOUS: User-Driven Access Control with Unmodified Operating Systems." In ACM Conference on Computer and Communications Security (CCS) 2016.

Smartphone (In)Security

Users accidentally install malicious applications.

Over 60% of Android malware steals your money via premium SMS, hides in fake forms of popular apps

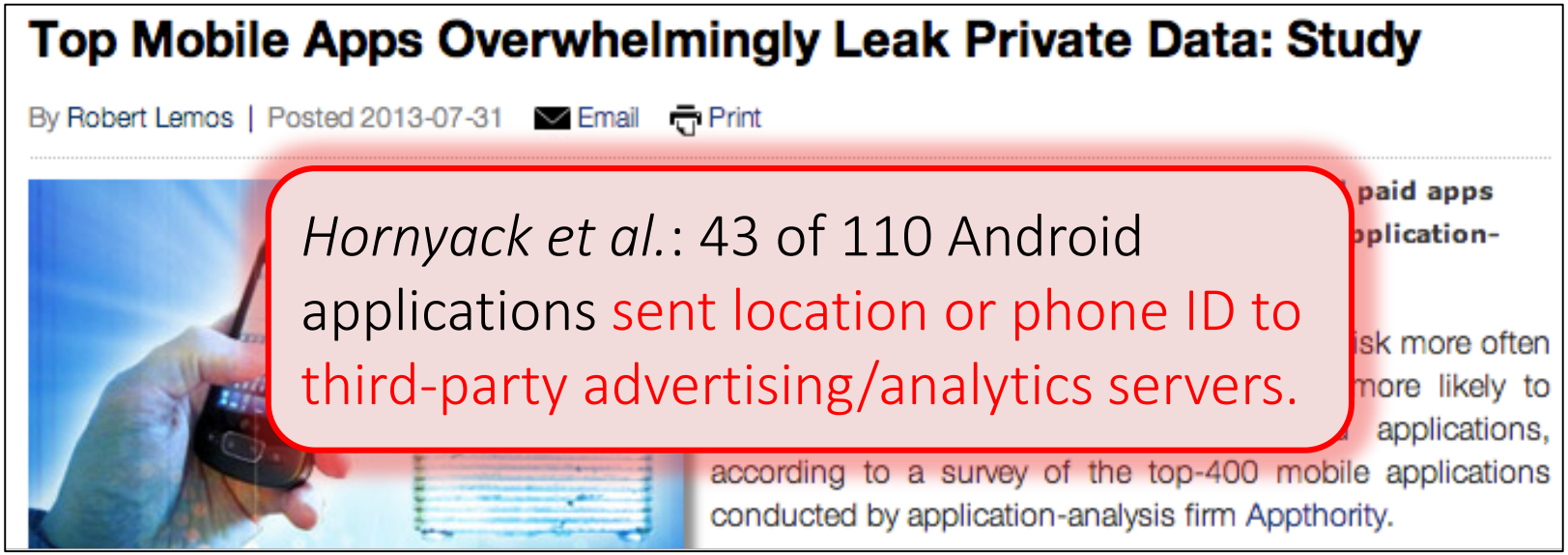
By Emil Protalinski, Friday, 5 Oct '12 , 05:50pm



Smartphone (In)Security

Users accidentally install malicious applications.

Even legitimate applications exhibit questionable behavior.



Top Mobile Apps Overwhelmingly Leak Private Data: Study

By Robert Lemos | Posted 2013-07-31 Email Print

Hornyack et al.: 43 of 110 Android applications sent location or phone ID to third-party advertising/analytics servers.

paid apps
pplication-

isk more often
more likely to
applications,
according to a survey of the top-400 mobile applications conducted by application-analysis firm Appthority.

Permission Granting Problem

Smartphones (and other modern OSes) try to prevent such attacks by limiting applications' access to:

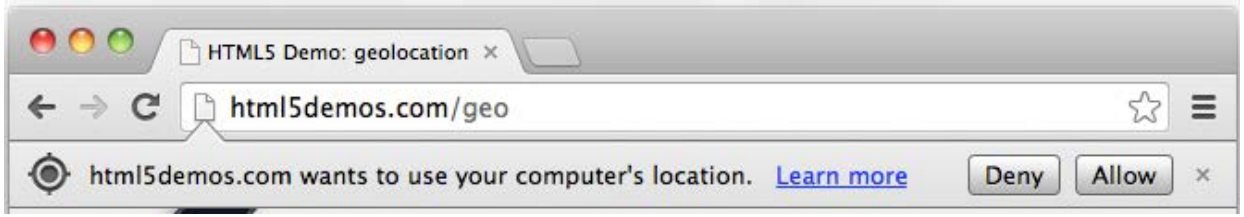
- System Resources (clipboard, file system).
- Devices (camera, GPS, phone, …).

How should operating system grant permissions to applications?

Standard approach: Ask the user.

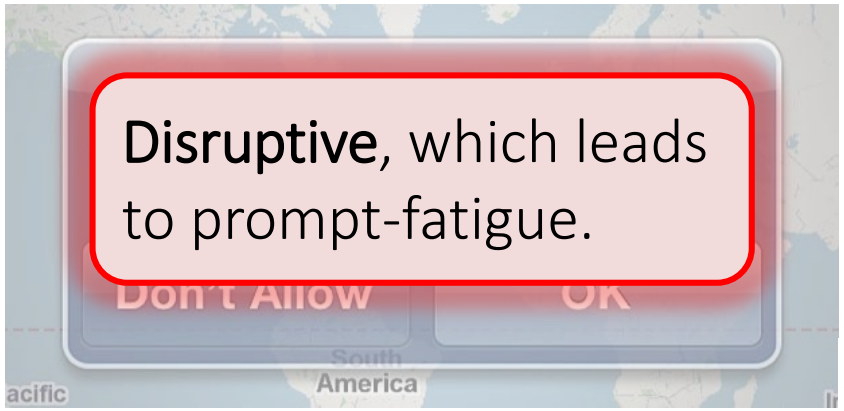
State of the Art

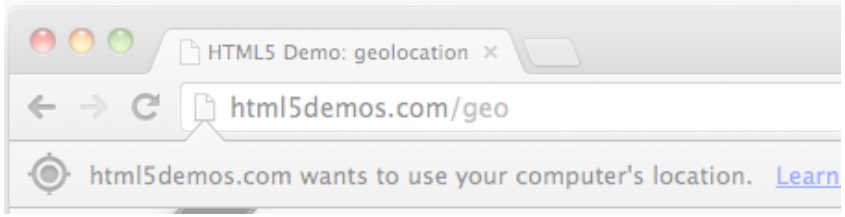
Prompts (time-of-use)



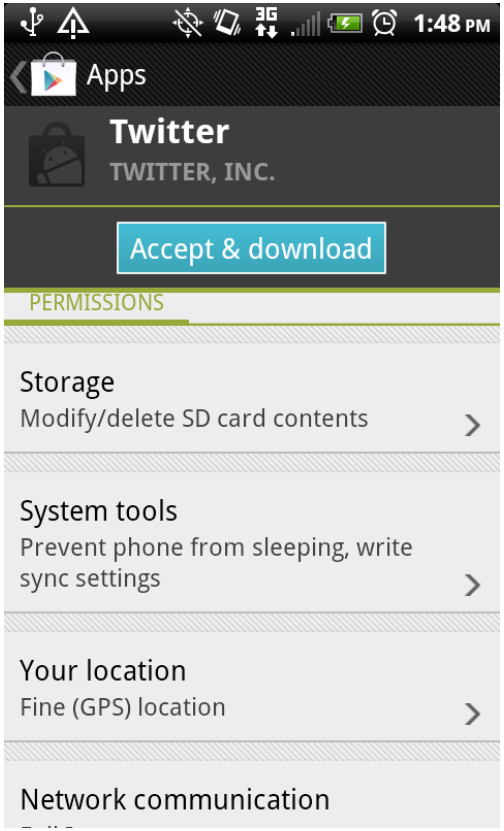
State of the Art

Prompts (time-of-use)



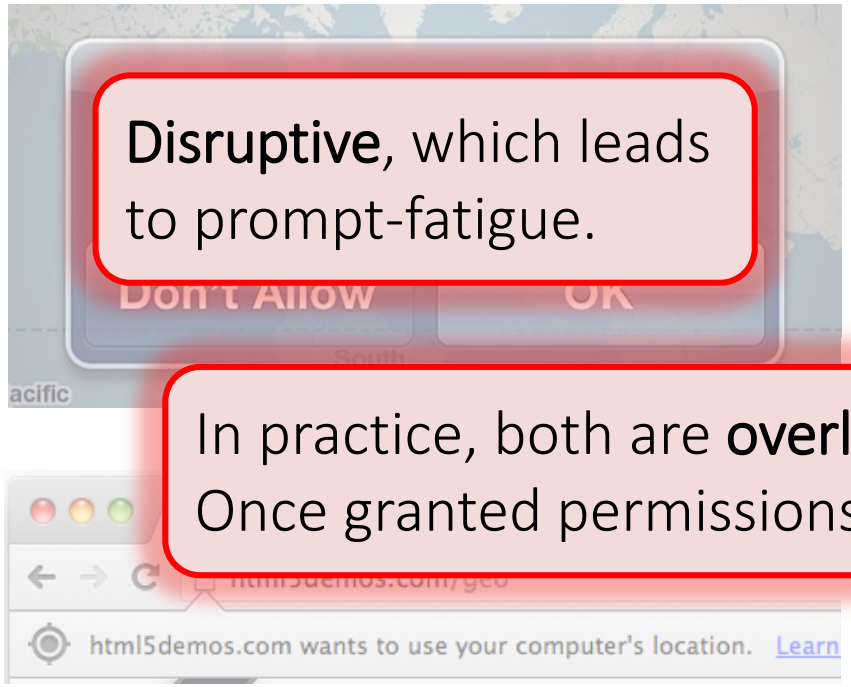


Manifests (install-time)



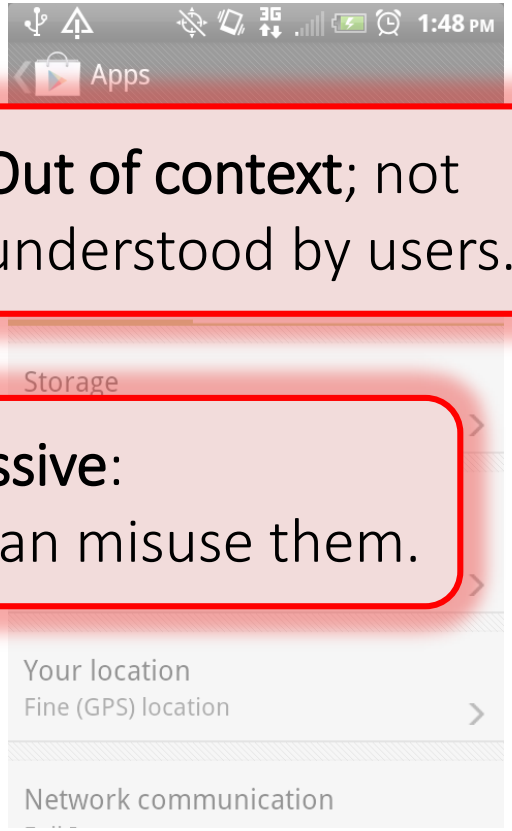
State of the Art

Prompts (time-of-use)



Disruptive, which leads to prompt-fatigue.

Manifests (install-time)



Out of context; not understood by users.

In practice, both are **overly permissive**:
Once granted permissions, apps can misuse them.

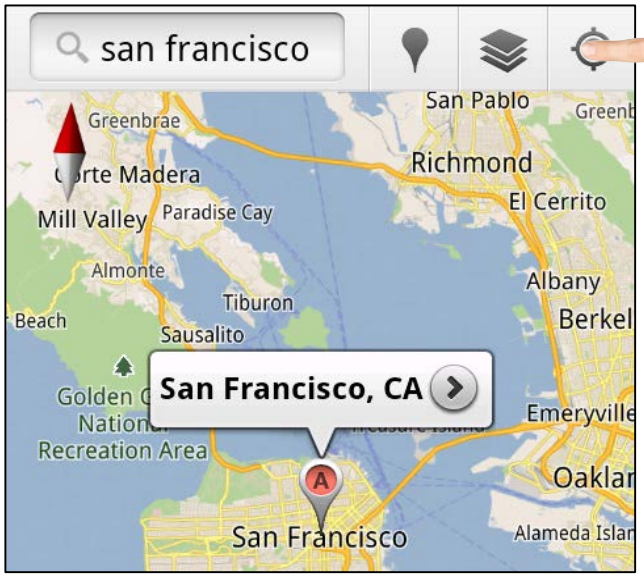
Goals for Permission Granting

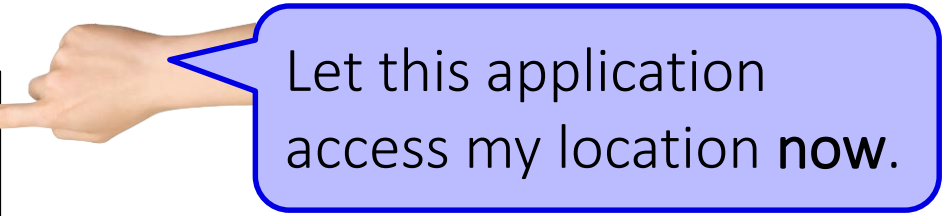
1. **Least-Privilege:** Applications should receive the minimum necessary access.
2. **Usable:**
   - Not disruptive to users.
   - Matches user expectations.
   - Doesn't require constant comprehension/management.

("magically" grants exactly those permissions expected by the user)

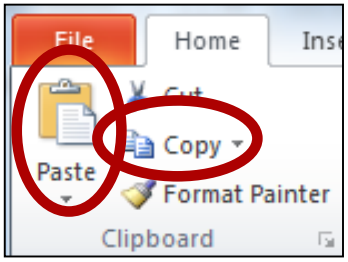
3. **Generalizable:** Easily extended to new resources.

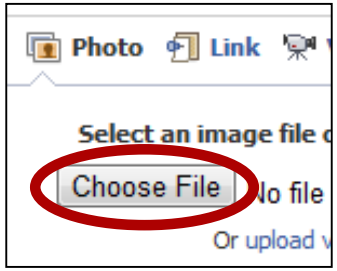
Our Work: User-Driven Access Control



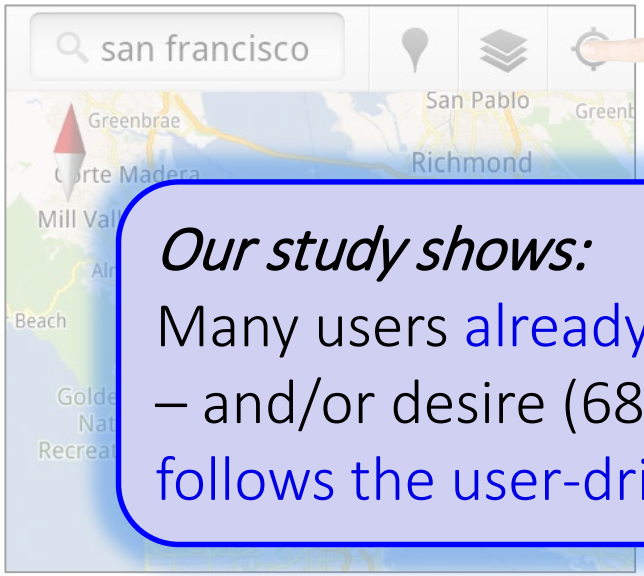


Insight:
A user's natural UI actions within an application implicitly carry permission-granting semantics.





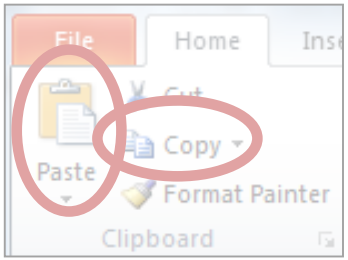
Our Work: User-Driven Access Control

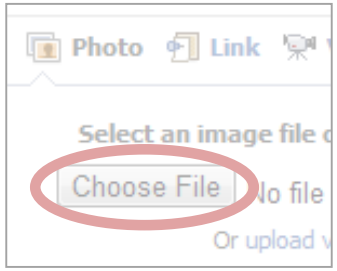


Let this application access my location **now**.

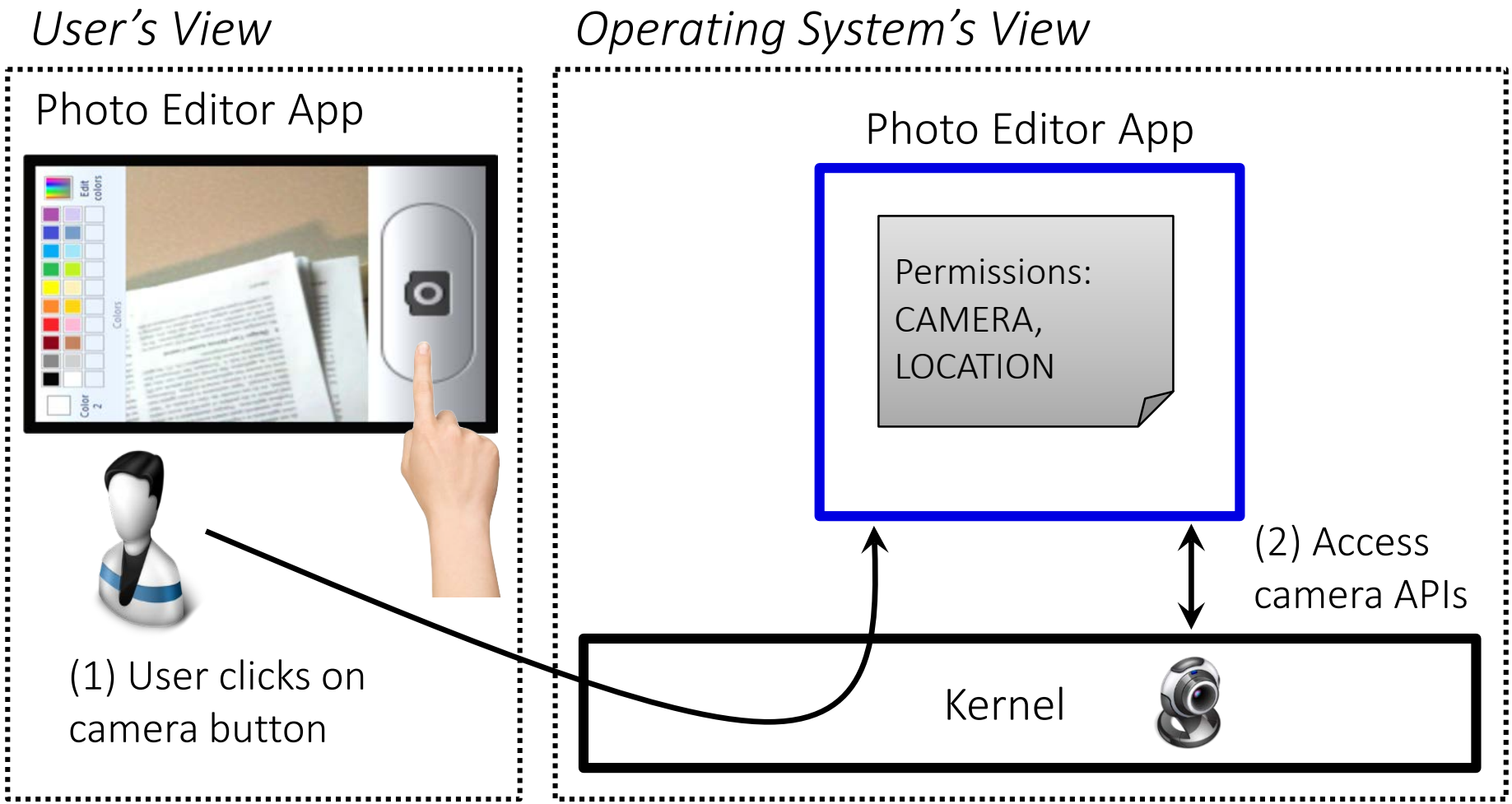
Our study shows:
Many users already believe (52% of 186) – and/or desire (68%) – that resource access follows the user-driven access control model.

ns within
carry
antics.

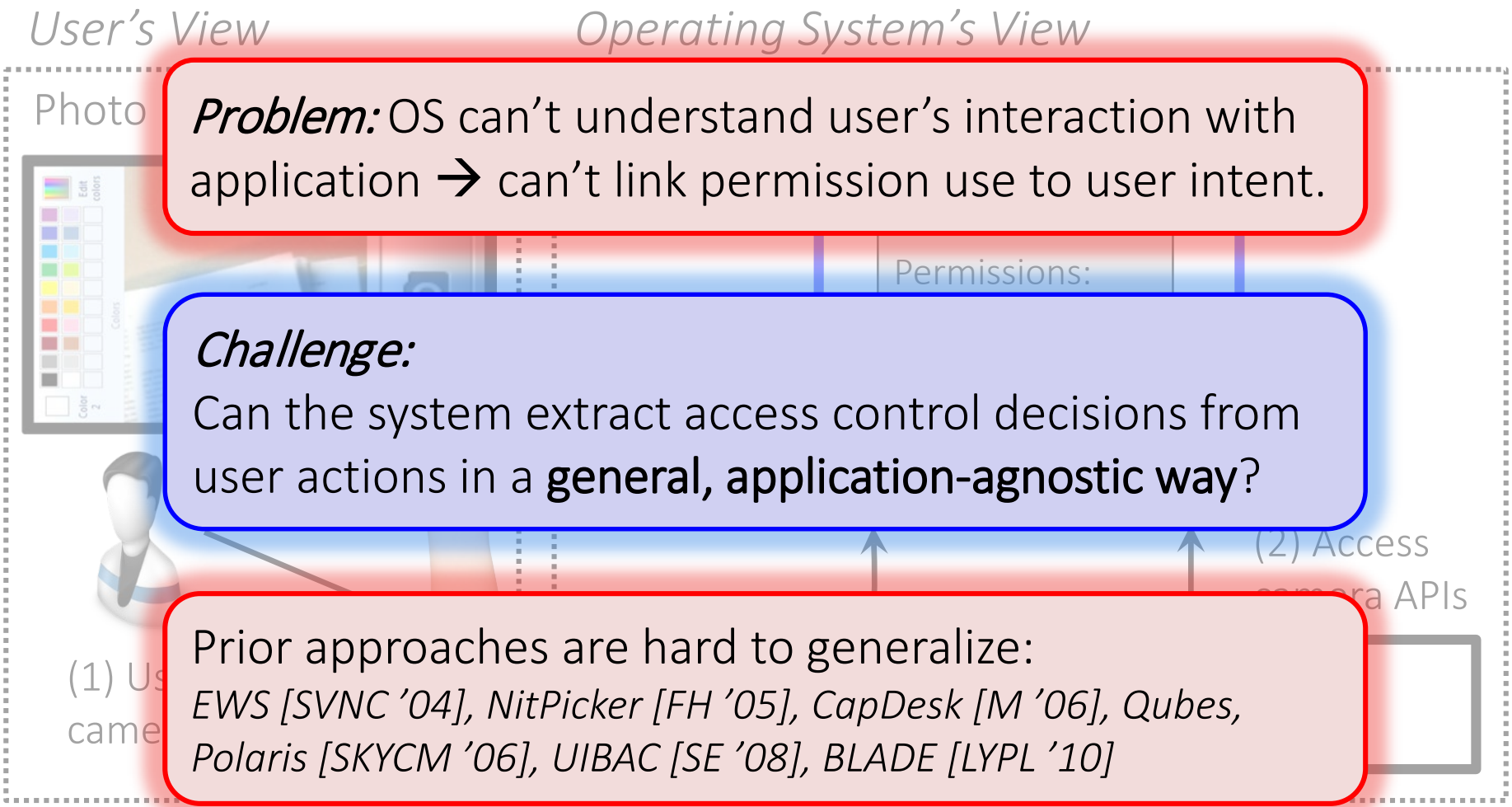




Resource-Related UIs Today



Resource-Related UIs Today

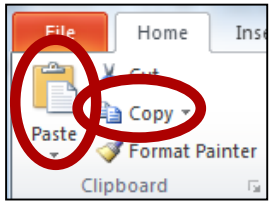


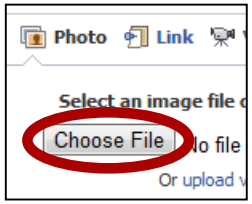
Problem: OS can't understand user's interaction with application → can't link permission use to user intent.

Challenge:
Can the system extract access control decisions from user actions in a **general, application-agnostic way**?

Prior approaches are hard to generalize:
EWS [SVNC '04], NitPicker [FH '05], CapDesk [M '06], Qubes, Polaris [SKYCM '06], UIBAC [SE '08], BLADE [LYPL '10]

New OS Primitive: Access Control Gadgets (ACGs)

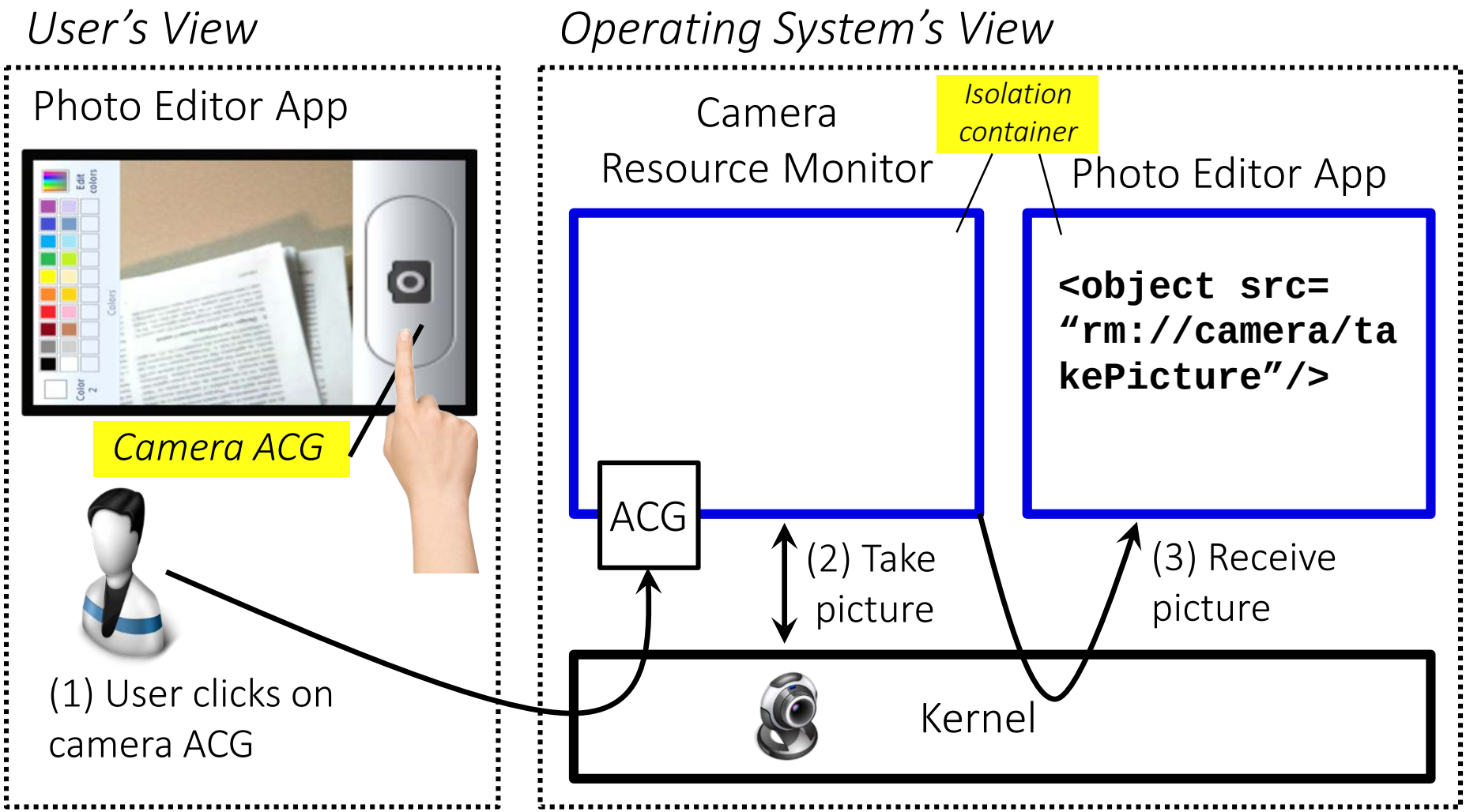




Approach: Make resource-related UI elements first-class operating system objects (access control gadgets).

- To receive resource access, applications must embed a system-provided ACG.
- ACGs allow the OS to capture the user's permission granting intent in application-agnostic way.

Access Control Gadgets (ACGs) in Action



Challenges with ACGs

Impact on applications:

- What about application customization?
- How to design system/resource APIs to support necessary application functionality?

Attacks on ACGs by malicious applications:

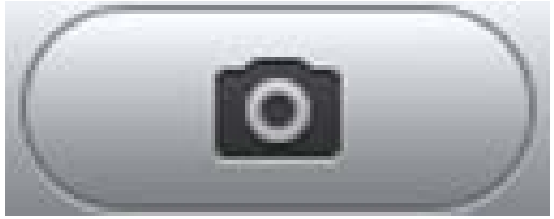
- How can system be sure that the user intent it captures is authentic?

Attacks on Access Control Gadgets

Malicious applications want to gain access without authentic user intent.

Example: Clickjacking attack.

Trick users into clicking on ACG by making it transparent.



Attacks on Access Control Gadgets

Malicious applications want to gain access without authentic user intent.

Example: Clickjacking attack.

Tric ent.

The operating system must protect ACGs from potentially malicious parent applications.

Start game!

First implemented in MSR's ServiceOS prototype system, later in Android (http://layercake.cs.washington.edu).

LayerCake: Secure Embedding for Android

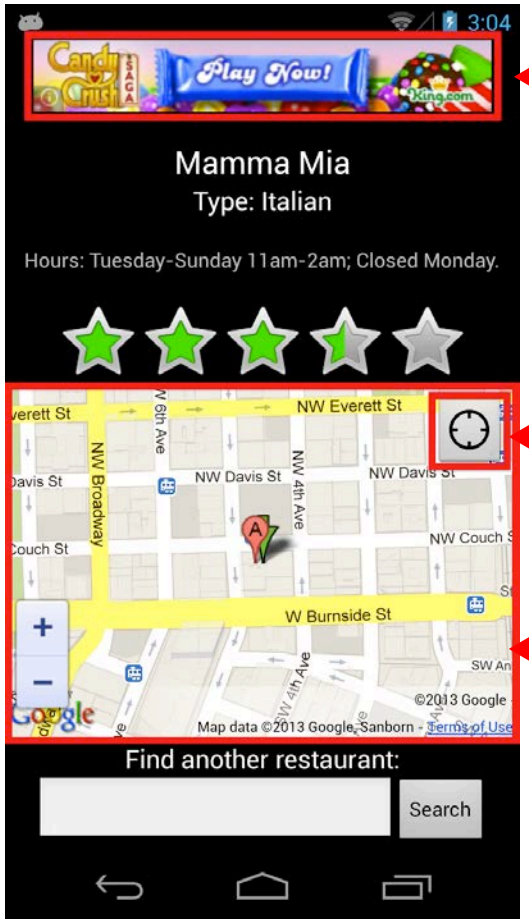
Modified Android 4.2 (JellyBean).

Goal: Allow an Activity in one application to securely embed an Activity from another app.

Pervasive changes to Android Window/Activity managers:

(1) **Separate processes.**

(2) **Separate windows.**



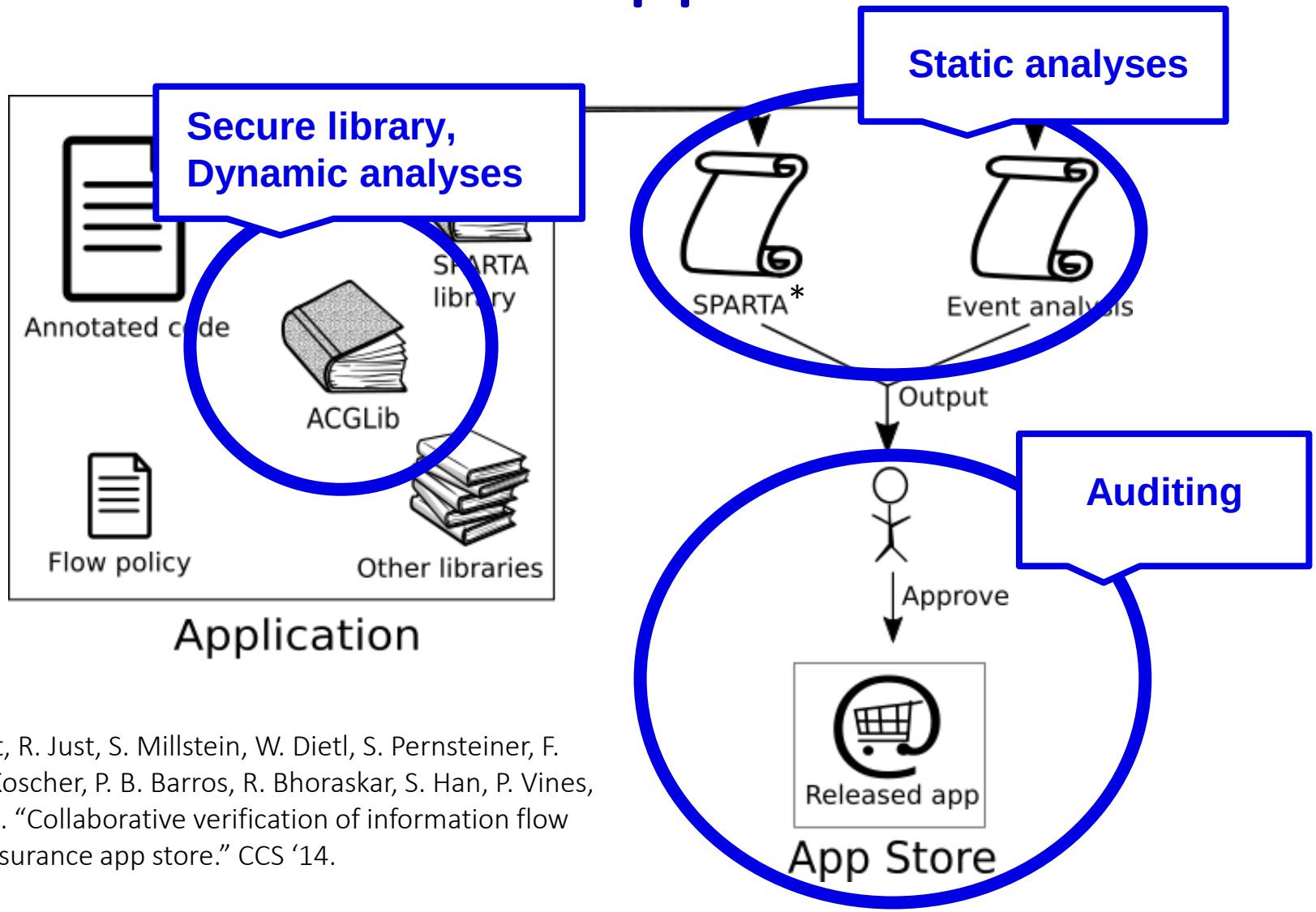
Ad Activity

Location ACG

Map Activity

[CCS '16]

UDAC without OS Support



* M. D. Ernst, R. Just, S. Millstein, W. Dietl, S. Pernsteiner, F. Roesner, K. Koscher, P. B. Barros, R. Bhoraskar, S. Han, P. Vines, and E. X. Wu. "Collaborative verification of information flow for a high-assurance app store." CCS '14.

Evaluation Highlights

User-driven access control matches user expectations.

Many users already believe (52% of 186) – and/or desire (68%) – that resource access follows the UDAC model.

User-driven access control improves security.

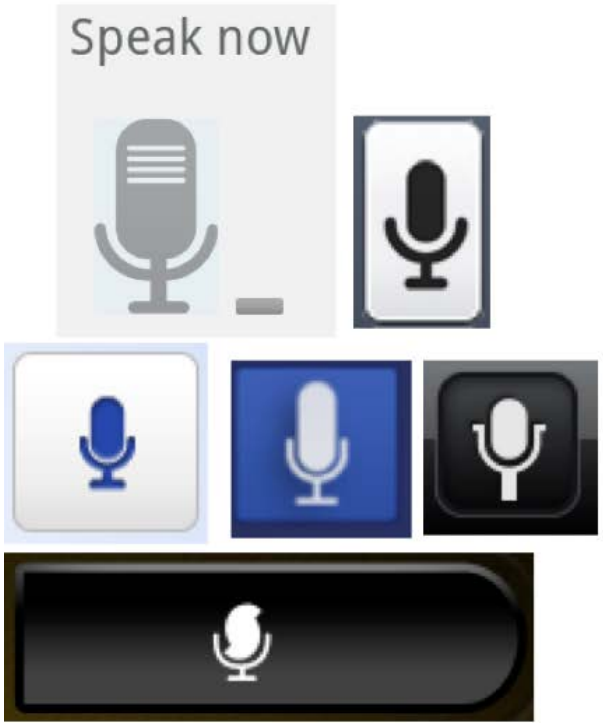
Addresses most published vulnerabilities related to resource access: 36 of 44 in Chrome (82%), 25 of 26 in Firefox (96%).

ACGs have minimal impact on user interface.

73% of top Android apps need only limited customization for resource-related UIs.

Evaluation Highlights

Limited Customization



Arbitrary Customization

73% of top Android apps need only limited customization for resource-related UIs.

Summary: Permission Granting

Prior approaches grant too much access, are too disruptive, or are not understood by users.

Our approach: user driven access control.

- OS extracts permissions from user actions.
- Enabled by new OS primitive: access control gadgets (must protect from malicious apps).
- Recent work: ACGs without OS support.
- Application-agnostic, improves security and matches user expectations.

Outline

I. The Web:
Third-Party Tracking

II. Modern OSes:
Permission Granting

III. Conclusion

Research Overview:
Improving Security & Privacy

franzi@cs.washington.edu

Thanks to many collaborators!


Analyze existing systems:
Web *[NSDI '12, USENIX Security '16]*, Automobilies *[IEEE S&P '10, USENIX Security '11]*, QR Codes *[MobiSys '15]*


Build new systems:
OS, Web, Smartphones *[IEEE S&P '12, CCS '16]*, UI Toolkits *[UIST '12, USENIX Seccurity '13]*, Usable encrypted email *[EuroS&P '16]*

Anticipate future technologies:
Robots *[HRI '15]*, Wearables, Augmented reality *[HotOS '13, CACM '14, CCS '14, HotMobile '16]*

Understand mental models:
Permissions, Journalists *[USENIX Security '15, PETS '16]*, Snapchat *[FC '14]*, Dev. world *[ICTD '16, DEV '16]*