Security Model for Web and Widgets

Security for Access to Device APIs from the Web - W3C Workshop
10 December 2008

Olli Immonen
olli.immonen@nokia.com

Nokia Research Center

NOKIA

Outline

- Browsing vs. widgets
- Identification, authentication, policies
- Declaring capabilities
- Attacks, risks
- Mashups
- Implicitly controlled access

- Exercise

Browsing vs. Widgets

Common thinking: browsing and widgets are different

- Browsing
  - Should be safe and fluent
- Widgets
  - Installation is a conscious decision
  - Could grant more powerful device access

But consider

- A widget that just has an entry point to a service. All code is fetched from web
- A browsing bookmark with user experience similar to that of a widget

Conclusion

- Maximally uniform solution for widgets and browsing

Identification

- Options
  - Site (2nd level (base) domain, full domain, URL)
  - DN from a certificate of a TLS server or code signature (widget or web page script)

- ... but
  - Users do *not* understand URLs even if they try to
  - PKI is even harder

- "This page" might be everything that the user understands

Authentication

- Options
  - No authentication, just DNS
  - SSL/TLS (https)
  - Signing (scripts of web pages or widgets)

- SSL/TLS could be helpful if enforced (TLS required for a certain API)
- Signing and identities are problematic in the open internet
- An approval type of signing not robust enough

- Opportunity
  - Widget developer's reputation in a community

Policy – How to Make It Visible to Users?

Option 1

- "Do you trust this site?"

  - Authorization less cumbersome
  - Risky APIs and combinations can be taken into account

Option 2

- "Do you allow this site to do X?"

  - Question is more concrete
  - Better aligned with POLA

Declaring Capabilities

- Established concept for applications (MIDP, widgets)
- Benefits
  - An entity can declare minimal capabilities (POLA!)
  - Even if the site fails (e.g. XSS) the damage is limited
- Declaration must be harder to change than the code (like XSS and eval())
  - Can be achieved by signing the rights declaration

- Would declaration be feasible for web browsing?
  - “Site security capability declaration“?
  - It might be tricky to declare the capabilities in advance

Attacks at Application and User Level

- User giving access unknowingly
  - The user gives access without understanding what is happening
  - Defense: Meaningful dialogues
- Impersonation
  - Similar to phishing
  - Defense: As against phishing (?)
- Vulnerable site
  - If the site the widget is accessing is vulnerable (XSS) then the device will be too
  - Defense: Grant only minimal access

Risk Assessment Concerns the Whole 'Value Chain'

Standards body	--- Specifications, security considerations
Vendor	--- Decision about enabling an API
Administrator	--- Setting the policy
End user	--- Trust decisions
Site, widget provider	--- Using 'risky' features (?)

Mashups

- Mashup web pages and widgets have content and code from multiple sources

- Challenges
  - Do all entities need to be identified, authenticated and given access rights separately or rely on one entity (page, widget)?
  - Enforcing access control
    - Setting permissions of components from various sources

Implicitly Controlled Access

- Available techniques
  - HTML form input e.g. **`<input type="file">`**
  - Special URL schemes e.g. **`mailto:`** , **`tel:`**
  - JavaScript APIs with a UI e.g. **`crypto.signText`**
- Possible 'APIs' (disclaimer: just examples)
  - Camera (take a picture)
  - Addressbook (select a person's email address)
- Benefit
  - Implicit authorization (selecting a particular file) is easier to understand than a question ("Do you allow the site to access your filesystem?").
- Drawbacks
  - Limits application UI design
  - Hardly feasible for features like continuous monitoring

Exercise: Thermometer API

Use case: Personalized product offerings depending on how warm/cold it is out there.

Continuous temperature monitoring is a privacy issue!

Just sending one measurement would be mostly OK.

Implicitly controlled method

- Thermometer UI with a button
- No separate access control

Continuous monitoring

- Authentication: just DNS would be OK
- “This site wants to monitor your temperature”
- “Trusted sites" should have the right …

Conclusions

- Uniform solution for device API access from widgets and browsing !
- Declaring rights is a good practice – could that be done for browsing, too
- Mashups require taking multiple entities into account
- Implicitly controlled access is an interesting option