WebVM

Security policy for device API access

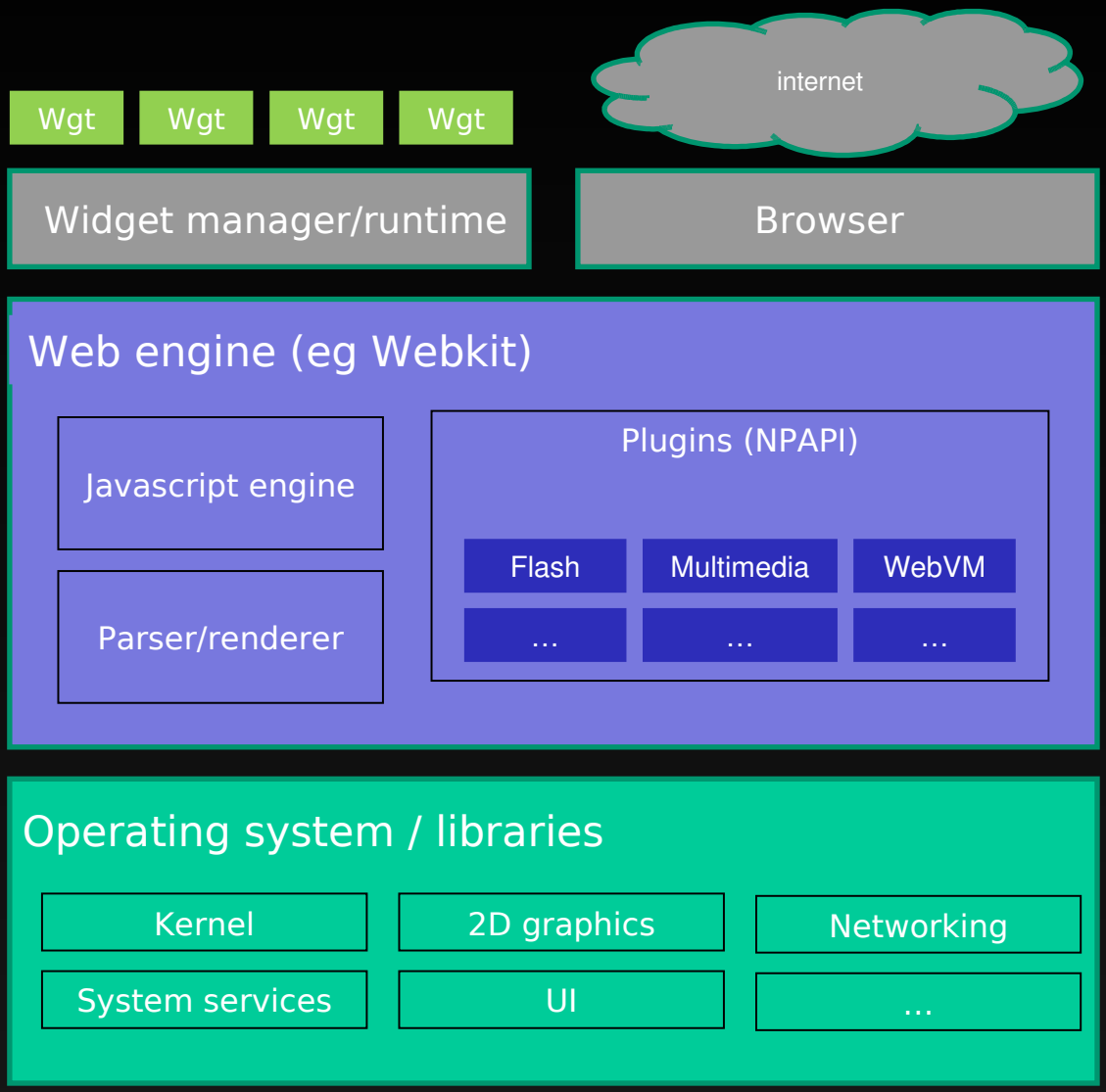
December 2008

Aplix Corporation

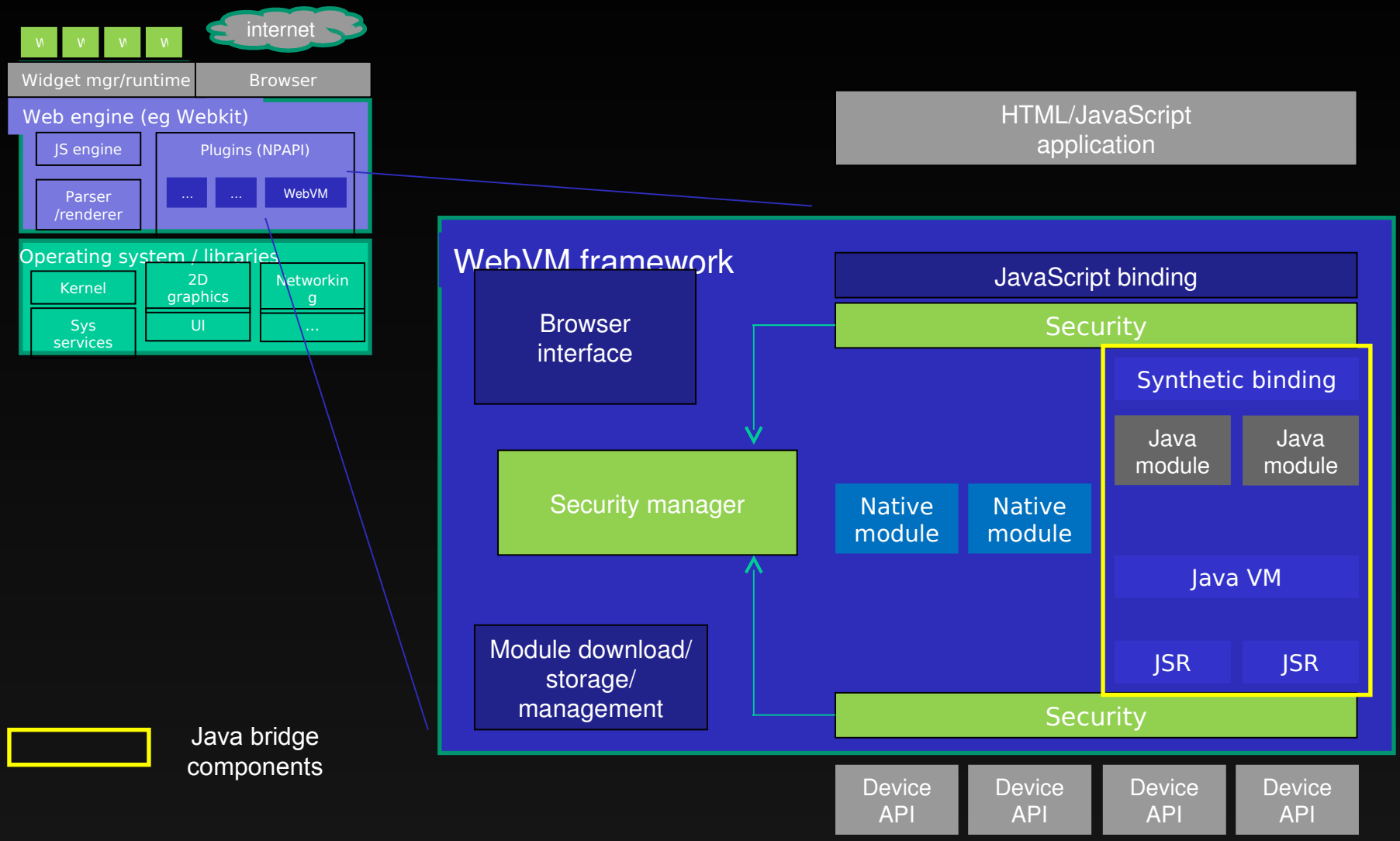
WebVM – in a nutshell

- a browser plugin
- provides a way for JavaScript programmers to get access to device APIs
- supports both websites and widgets
- does not by itself define any APIs
  - web applications identify the APIs they wish to use explicitly
- supports the implementation of specific APIs
  - natively (ie in C / C++)
  - in Java
- the Java “bridge” is interesting
  - most phones already have a Java environment exposing a significant number of device APIs
  - implementations of APIs can be added or upgraded dynamically
  - access to sensitive device features are uniformly and securely mediated by an access control framework

Browser architecture



WebVM architecture



Security objectives and features

- WebVM includes a policy-driven access control framework that governs access to device APIs
- Aims
  - accommodate multiple trust models
  - support fine-grained access control
  - uniformly deal with websites as well as widgets

Multiple trust models

- many mobile application models confuse authenticity and trust
  - signature on widget package establishes authenticity of package
  - … but trust model often assigns trust based on the root cert
  - places determination of trust with the CA, not with the user
  - model does not scale
- alternative model uses signature to establish authenticity, but trust is determined separately, eg
  - by user
  - party to which user has delegated authority
- we make no assumption about the specific trust model
  - subject attributes exposed to include end-entity and root cert attributes
  - for widgets, may support multiple signature profiles

Fine-grained access control policy

- A prerequisite for effective policies
  - those that can accurately discriminate between legitimate and unwarranted requests
- Fine-grained subject attributes
  - can express rules at the level of broad trust domains or individual sites or widgets
- Fine-grained resource attributes
  - can express rules at the level of groupings of APIs, individual device features, or specific parameters
- Combination of rules and different effects
  - eg user-defined “deny” rule can override operator-defined “permit”

Multiple identity systems

- A single framework supports websites and widgets
- Each has its own system of identities (subject/subject attributes)
- Website
  - protocol, port, host
  - signer DN if jar:
  - identity of containing page determines rights of contained iframes
- Widget
  - id (uri)
  - end entity cert attributes
  - root cert attributes
  - multiple signatures and signature profiles
- Policies can define “trust zones” containing identities of each type

Trust model for API implementations

- WebVM allows
  - app to call independently implemented API
  - … which in turn attempts security-relevant operation on platform
  - question: who is considered to be attempting that operation?
- “Pass-through” security model
  - All events are considered to be attempted actions by the containing page
- “Trusted subsystem” model
  - WebVM library requires access to specific platform APIs
  - Exposes a higher-level service to invoking web applications
  - Is trusted not to expose the full generality of those platform APIs to the web app
  - requires the WebVM library to be signed, verified, trusted, and installed

Access control policy model and language

- XACML-inspired model
  - policy set is tree of policies, with combining rules
  - policy has a target and contains rules
  - rules have a condition and effects
- Differences from XACML
  - some optimisation-driven reduction in generality
  - some extensions motivated by environment and use cases
    - support for “undefined” values
    - additional combining rule
    - additional effects involving prompts
  - more natural and compact XML representation
- Definition of model, language, attribute dictionaries, contributed to BONDI

Summary

- WebVM attempts to accommodate multiple styles of access control policy within a single framework
- If standardisation of the policy model and language are contemplated
  - should not “hard-code” a specific trust model into the standard
    - must expose sufficiently many subject and resource attributes to implement reasonably envisaged policies
  - should be capable of fine-grained control to ensure policies are effective
  - the significance of a signature must be explicit
- The configuration problem still needs to be addressed
  - examining remote provisioning of policy fragments and delegated authority in BONDI
- Still some open issues for websites
  - these are also problems for website security generally