The *webinos* Project

Christian Fuhrhop
Fraunhofer FOKUS
Kaiserin-Augusta-Allee 31
10589 Berlin, Germany
christian.fuhrhop@
fokus.fraunhofer.de

John Lyle and
Shamal Faily
Department of Computer
Science, University of Oxford
Oxford, UK
first.last@cs.ox.ac.uk

ABSTRACT

This poster paper describes the webinos project and presents the architecture and security features developed in webinos.

It highlights the main objectives and concepts of the project and describes the architecture derived to achive the objectives.

Categories and Subject Descriptors

C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*distributed applications*

Keywords

Cross-domain and cross device development, car, home media, PC, mobile, web runtime, application platform, open source, security, privacy, personal zone

1. INTRODUCTION

The webinos project has a duration of three years (from September 2010 to August 2013) and is co-funded by the EU under the FP7-ICT-2009-5 Programme - Objective 1.2.

The project is building an application platform featuring web components that can be deployed on mobiles, PC, in-car units and TVs supporting a web runtime, which will support the rapid creation of innovative and secure web applications, where:

- applications can be used in several contexts across all user devices,
- applications can make use of the specific capabilities and resources that the underlying hardware platform provides,
- applications can securely access private and non-private data and other services on the cloud and social web, as well as data on the user terminal,
- applications are developed once by service providers and deployed anywhere,
- manufacturers and network operators have a service platform constituted by open standards and open source, which satisfies the requirements of the four domains mobile, PC, Home media and in-car units.

Copyright is held by the International World Wide Web Conference Committee (IW3C2). Distribution of these papers is limited to classroom use, and personal use by others.
WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

2. PROJECT OVERVIEW

webinos envisions web applications running and utilizing resources across a range of connected devices, interacting seamlessly with each other to contribute to the development of the internet and facilitate users with more attractive, innovative and valuable applications.

To achieve this, webinos will define and deliver an Open Source platform which will extend existing Web technologies to enable Web applications and services to be used and shared consistently and securely over a broad spectrum of connected devices, including mobile, PC, home media (TV) and in-car units. This platform will have a concrete implementation that is accessible to all as an Open Source asset.

2.1 Project objectives

The primary objectives of the webinos project are to:

- Deliver a secure Open Source platform based on web technologies in the mobile, home media (TV), PC and automotive domains.
- Define and build key enablers where such are not available, to facilitate the efficient and cost effective deployment of applications to converged devices.
- Define and build a web based application security framework that addresses user and service provider needs, underlying platform and device requirements and delivery and management aspects, which has been technically proven and is easily deployable.

2.2 Project highlights

The webinos project supports the following key concepts:

- webinos builds on the achievements of the Web community and extends an open source web runtime environment.
- webinos is a federated web runtime that offers a common set of APIs to allow apps easy access to cross-user, cross-service, cross-device functionality, in an open yet secure manner.
- webinos creates open specifications and Open Source reference implementations that not only show the feasibility of specifications but also simplifies the adaptation by the industry.
- webinos allows discovery of devices and services not only based on technical, but also on contextual information, such as social proximity. The use of social

proximity will also be used to help users find adequate privacy preferences by aligning them with decisions made by trusted friends.

2.3 Primary innovations (so far)

To achieve the goal of providing an open, secure, distributed and cross-domain platform for the easy deployment of web applications, webinos introduces the following three primary innovations:

2.3.1 Personal Zones

Devices are registered in Personal Zones, which act as Overlay Networks and abstract from the underlying physical networks and protocols.

The Personal Zone provides simple access to local and remote services, simplifying the task of an application programmer, allows simple discovery of devices and services, provides adaptation based on access context and supply communication paths based on trust relationships, decoupled from underlying technologies.

2.3.2 Personal Zone Proxy

A webinos Personal Zone Proxy acts as stand-in for Messaging, Discovery and Security functions when a device has no Internet connectivity. It stores information (events, messages) and synchronises with a central hub once connectivity is re-established, thus reducing the 'housekeeping' requirements of application developers.

Additionally, the Personal Zone Proxy acts as the policy enforcement point for the access to device APIs and is responsible for local discovery and connectivity to non-IP based devices, such as Bluetooth, ZigBee, NFC and locally connected devices.

2.3.3 Security Framework

As a secure web platform, webinos started with security concepts developed for the WAC platform and extended them for use in a distributed environment. The most significant feature is the security policy architecture, which primarily controls applications' access to device features, but also states rules about inter-device communication and event handling.

3. ARCHITECTURE COMPONENTS

Users own and use an increasing number of devices. To simplify the management of those devices and the services that are used on and by them, webinos uses the concept of a 'personal zone'. [1].

Devices within a personal zone are authenticated to the zone, avoiding the need to establish direct peering relationships between individual devices pairs.

The zone also provides a shared model of the context, providing information about users, device capabilities, properties and the environment, allowing the synchronization of devices in the zone in regards to authentication, personal preferences and service specific data. Since not all devices will be continously online, the synchronization also is essential for supporting offline usage.

The personal zone also provides discovery of services and access to discovered services, including the access to services that may only be accessible via intermediate devices, for example on Bluetooth devices that can only be directly reached by devices that are in physical proximity.

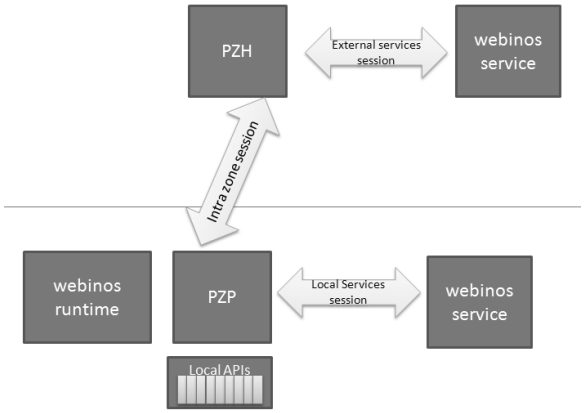

Figure 1: Main webinos architecture components

Personal Zone Hub (PZH)

The Personal Zone Hub (PZH) is the primary element to handle the requirements to establish a personal zone.[1].

Technically, a PZH is a service and is identified by a URL and supports a RESTful API based upon JSON RPC. It enables access the Zone's devices and services for the duration of a browsing session. It also enables discovery of and access to devices access by others, subject to the policies that the owner of the zone has defined.

A PZH can subscribe to near instant notifications when a topic (feed URL) the user is interested in is updated.

The PZH also provides support for discovering other hubs based upon someone's full name or pseudonym. This is implemented as a federated discovery process across hubs, starting from the personal hub. The process is trusted with access to personal data for ranking purposes, but is designed to avoid disclosing such data, except as permitted by the owner's policies.

Personal Zone Hubs can also be discovered starting from someone's email address or phone number. The email addresses domain name can be used to locate a query service (typically provided by the domain owner). Note that users may choose to limit discovery, e.g. to people within a given group, or to prevent discovery altogether, in which case it is up to the user to communicate the URL for their Personal Zone Hub to others as needed.

Personal Zone Proxy (PZP)

A Personal Zone Proxy (PZP) is running on every device and is responsible for the communication with the PZH.[1].

If access to the PZH is not available, it is the responsibility of the PZP to act in its place and store relevant information and events, synchronizing them with the PZH when contact is re-established.

In addition to acting as a proxy to the PZH, the PZP is responsible for all discovery using local hardware based bearers.

Finally, since the external interface to the PZP is defined, a pseudo-PZP can be used to enable webinos access to devices that cannot implement a full webinos runtime (due to device limitations or legal reasons), but can use the PZP interface to expose access to services available on that device.

4. WEBINOS DEVICE APIS

Based on a study of scenarios, use cases and requirements in the first half year of the project, webinos determined a set of needed device APIs. If existing APIs matched the requirements of the project, these APIs were used. If there were specific needs that were not already covered by existing APIs, the attempt was made to stay as closely to existing APIs as possible, making modifications and extensions with as little disruption to the original APIs as possible.[2].

The APIs provided by webinos are: Attestation, Authentication, Context, Events, AppLauncher, Messaging,NFC (near field communication), Payment, Sensors, Discovery, TV, Userprofile, Vehicle, WebinosCore, Widget, Devicestatus vocabulary, W3C calendar (*), W3C contacts (*), WAC devicestatus (*), WAC deviceinteraction (*), W3C DeviceOrientation Event specification (*), W3C File, File Writer, Directories and System (*), W3C Gallery (*), W3C Geolocation (*), W3C Media Capture (*) *(APIs with an asterisk denote unmodified, referenced APIs)*

The application executing on top of the runtime will access the API via a JavaScript binding. However, this will not directly access the underlying device functionality, but communicate the request to the PZP via a JSON RPC call. The PZP is implemented as node.js code, providing the actual access to the device features and services.

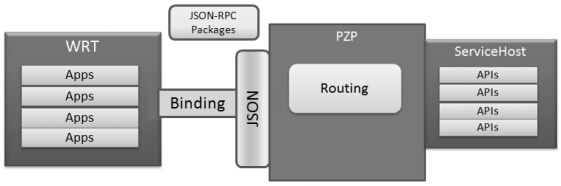

Figure 2: Access to APIs via PZP

This approach provides the following advantages:

- All API calls pass through a single interface, which can then beAttestation used as a Policy Enforcement Point (PEP) by the policy and security framework, mediating access to the APIs.
- Since a RPC interface is defined for all API functions, provideing access to APIs on remote devices is not significant different from access to local APIs, aiding device interaction and interoperability.
- Since the device specific code is primarily in the PZP and the communication to it is based on standard protocols, the need to extend the browser or web runtime is diminished, aiding portability of the webinos environment.

5. WEBINOS SECURITY ARCHITECTURE

webinos has been designed with security and privacy as primary goals. There are a number of privacy concerns: webinos is likely to hold personal information and support access to sensitive APIs such as geolocation, calendars and contacts. A comprehensive access control system and intuitive privacy controls are therefore essential. From a security perspective, webinos may be considered a lucrative target for attack due to the support for a payment API and the storage of user credentials to access online services. webinos may also be used in a corporate environment, meaning that valuable company IPR could be present on a webinos device. As a result, webinos must protect the integrity of the platform and provide secure storage of user and application data.

Additionally, webinos aims to become an ubiquitous software platform, much like the web browser. This means that a successful attack on the system could potentially affect all devices using it. This, in combination with the inevitable complexity of such systems, is known as the *middleware problem* [3]. Another problem requiring novel privacy controls is the management of *context*: information about a user's location, activities, nearby devices and social graphs. This can bring huge benefits, adapting the user experience depending on their environment, as well as creating new connections between friends and local services [4]. However, there is a tension between the use and storage of contextual data and personal privacy – how can users gain the benefits of one without losing the other?

The webinos project has tackled these issues through two activities: a study of user expectations of security and privacy and the development of a comprehensive security architecture.

The study of user expectations involved carrying out several security and usability analysis activities. Several *personas* [5] were created to describe archetypical behaviour that target webinos users might carry out. These personas were derived from a variety of different data sources and contextualised using scenarios for the different social contexts where webinos applications might be used. They represent end-users, application developers, and people who, while not end-users of webinos, are likely to be affected by it. These perspectives were important because, unlike traditional middleware which is usually exposed only to application developers, end-users are expected to assert their own security and privacy expectations when managing their personal zones.

As well as informing usability considerations of webinos, personas were also developed to envisage the different ways webinos might be attacked. These *attacker personas* were grounded in open source threat analysis data [6, 7], and published case histories on convicted hackers [8]. To validate the attacker personas, misuse cases [9] were created to explore whether these attackers have both the capabilities and motivation to exploit different aspects of the webinos platform.

The resulting security architecture can be divided into the following overlapping areas:

5.1 Personal zone communication

Each personal zone hub will act as a certificate authority for the zone, issuing certificates for individual proxies. Communication between any device on the zone will then be mutually authenticated and encrypted using standard Transport Level Security. The personal zone hub will make sure that each device has a copy of all known certificates, so that peer-to-peer communication is supported. In the event that a device is lost or stolen, the hub can be accessed remotely to revoke certificates and remove it from the zone.

5.2 Inter-zone communication

webinos devices in one personal zone will be able to com-

municate with those in other zones through a combination of certificate exchange and device pairing mechanisms to ensure that identities are established securely without relying on any trusted third parties.

5.3 Access control policies for applications

webinos applications will, in line with current standards, require explicit permissions to access the device resources, but takes this further by integrating privacy policies which govern how applications promise to use data they request access too. After permissions are granted, access control rules are automatically generated in XACML. These can then be extended to include rules about sharing data between applications, users and devices. webinos will synchronise policies between all devices in the personal zone so that users do not have to make the same decisions repeatedly.

5.4 Runtime protection

The webinos platform is being designed with robustness in mind. This means following best practices in secure development, as well as taking steps to ensure that each software instance is not compromised. This may be achieved through the *attestation API* which can report to trusted parties (such as the PZH) whether or not a webinos device is running software with the latest patches installed. Runtime components will also be isolated from each other as much as possible.

5.5 Secure storage

Several types of stored data in webinos require protection, including access control policies, credentials, keys and user context data. webinos will take advantage of local device features to encrypt this data, minimising the cost of a mobile device being lost or stolen. [10]

6. CONCLUSIONS AND OUTLOOK

In its first half year, webinos has been studying the requirements for a web runtime that supports cross-device, cross-domain applications, is secure by design, can be provided as Open Source software and is highly portable.

In the second half year, webinos specified a system based on those requirements, defining a basic architecture, key elements to enable the requirements and a set of APIs.

The next half year of webinos will be dedicated to provide an implementation of the webinos system and the provision of example, proof of concept and showcase applications, before re-evaluating and refining the system in the second phase of the project.

7. ACKNOWLEDGMENTS

About a hundred people from 24 companies have been involved in the webinos so far, making the list too long to include them here. Their valuable contributions to the project are hereby acknowledged in toto.

This project is carried out with financial support from the European Community under the FP7-ICT-2009-5 Programme - Objective 1.2.

8. ADDITIONAL AUTHORS

As webinos is a highly cooperative project, this poster paper is based on project material provided by Dave Ragett (W3C, email: `dsr@w3c.org`) (webinos architecture), Nick Allott (Impleo, email: `nick@nickallott.com`) (key architectural components), Claes Nilsson (Sony Ericsson Mobile, email: `claes1.nilsson@sonyericsson.com`) and Katrin Jordan (Deutsche Telekom, email: `katrin.jordan@telekom.de`) (project factsheet).

9. REFERENCES

[1] webinos Project. Deliverable 3.1: Phase 1 architecture and components, June 2011.

[2] webinos Project. Deliverable 3.2: Phase 1 device, network, and server-side api specifications, June 2011.

[3] A. Cooper and A. Martin. Towards a secure, tamper-proof grid platform. In *Cluster Computing and the Grid, 2006. CCGRID 06. Sixth IEEE International Symposium on*, volume 1, page 8 pp., may 2006.

[4] George Gionis, Heiko Desruelle, Dieter Blomme, John Lyle, Shamal Faily, and Louay Bassbouss. 'do we know each other or is it just our devices?': A federated context model for describing social activity across devices. In *W3C/PrimeLife Federated Social Web Europe Conference 2011*, June 2011.

[5] John Pruitt and Tamara Adlin. *The persona lifecycle: keeping people in mind throughout product design.* Elsevier, 2006.

[6] Open Web Application Project (OWASP) web site, January 2011.

[7] The MITRE Corporation. Common Attack Pattern Enumeration and Classification (CAPEC) web site, August 2011.

[8] Andrea Atzeni, Shamal Faily, John Lyle, Cesare Cameroni, and Ivan Fléchais. Here's Johnny: a Methodology for Developing Attacker Personas. In *Proceedings of the 6th International Conference on Availability, Reliability and Security*, pages 722–727, 2011.

[9] Guttorm Sindre and Andreas L. Opdahl. Eliciting security requirements with misuse cases. *Requirements Engineering*, 10(1):34–44, 2005.

[10] webinos Project. Deliverable 3.5 : Phase 1 security framework, June 2011.