Galois Proposed Project 1

Mobile Verifiable Voting

Design and develop a remote verifiable voting application that runs on iOS or Android.

Problem Description and Project Purpose

An internet-based end-to-end verifiable voting [1] prototype is being developed at Galois. The system implements a variant of a peer-reviewed system called Remotegrity [2].

Voting can be performed by voters using either a plain old HTML interface or using client-side cryptography. Voters, observers, and election officials can perform vote and election verification using client-side cryptography. The system is meant to be software independent [3] and high-assurance [4] (i.e., a high EAL level).

Our example client is written in the Haskell functional programming language [5], targets the Windows, OS X, and Linux desktop OSs, and performs client-side cryptography. Porting this client to either iOS or Android would involve using our crypto code as a library and designing and developing only the native mobile UI/UX workflow.

Developing a verifiable elections mobile application for demonstration purposes would have high impact on the future of verifiable elections in the USA and abroad.

Actions and Deliverables

- Learn the basics of remote verifiable election schemes.
- Make a low-tech mobile mockup of the election workflow for either or both voting or vote/election verification based upon the existing system.
- Design a storyboard in the Android IDE or Xcode for the mockup.
- Implement the storyboard using the Galois election crypto library.
- Document the system design and development from a developer and a user point-of-view.

[1] http://en.wikipedia.org/wiki/End-to-end_auditable_voting_systems
[2] https://eprint.iacr.org/2013/214.pdf
[3] http://en.wikipedia.org/wiki/Software_independence
[4] http://en.wikipedia.org/wiki/Evaluation_Assurance_Level
[5] https://www.haskell.org/

Resources and Support

Galois will provide remote deep technical R&D assistance on matters relevant to our expertise.

Expertise Necessary

At least one student on a team executing on this project must have UI design experience. At least one student on the team must have mobile application development experience or be willing to learn about such quickly. (E.g., do the iTunes University course on iOS development from Stanford. [6]) The client can be implemented in Haskell if the developers already have expertise with such, otherwise they would use Java, Objective-C, or Swift.

[6] https://itunes.apple.com/us/course/developing-ios-7-apps-for/id733644550