Enforcing Information Flow Requirements in Web Applications
Michael Paulauski
advised by David Naumann and Andrey Chudnov

- Distributed applications involve sharing of sensitive data
- Policies pertain to flows of information, which goes by many channels (data in network, data in memory, timing, visual cues to users, ...)
- Encryption protects data in transit, but it is processed in memory on computers running many different programs, some malicious
- Goal: strong guarantees about behavior of those programs: absence of bugs or trojan code that leaks or corrupts sensitive information — yet allowing useful cooperation, e.g., within web mashup
- Challenges: it is difficult to specify what flows of info are allowed, precisely enough to support
- Rigorous validation for high assurance (e.g., for financial and military applications). Difficult to predict behavior of programs
- Our research team is developing tools for automated software validation that can also detect implicit flows of information via control flows of code

- PhD Student Andrey Chudnov is developing an in-browser information flow monitor for JavaScript. Case studies on realistic web applications are necessary to evaluate effectiveness and permissiveness
- One possible case study is 'Anatomy ProAm' - research prototype of a Facebook game crowdsourcing interpretation of CT scans. However, it's very complex. We need a simpler application for an initial case study
- This project’s goal: to develop a simple mashup web application with interesting security policies

**Application Workflow**
- The user enters the specific UPS tracking number(s) of the desired package
- My code then takes this tracking number, establishes an HTTPS (SSL) connection with UPS’s servers, and requests XML data for the designated package’s route
- Once my website receives the XML data, it parses and extracts the relevant route data
- My website then passes this route data to Google Maps, which first translates street addresses to GPS coordinates and then plots those coordinates onto a Google Map object built into the website
- Finally, my website links these points on the map together with a solid line indicating the package’s route and prints out a detailed text-based list of the completed route

**Challenges**
- Developing complete mashup from scratch
- Complicated program components/development setup
- Need to use a proxy server to work around the Same Origin Policy

**Examples of legitimate and illegal information flows in the medical information setting**

**Representative publications**
Bannister, Naumann and Rosenberg

Tracking Information Flow in Dynamic Tree Structures, 14th European Symposium on Research in Computer Security, 2009
Russo, Sabelfeld and Chudnov

Chudnov and Naumann

**Funding**
National Science Foundation (REU), Stevens Scholars Summer Research Program

![Diagram showing information flows and policies in the package tracking mashup](image-url)