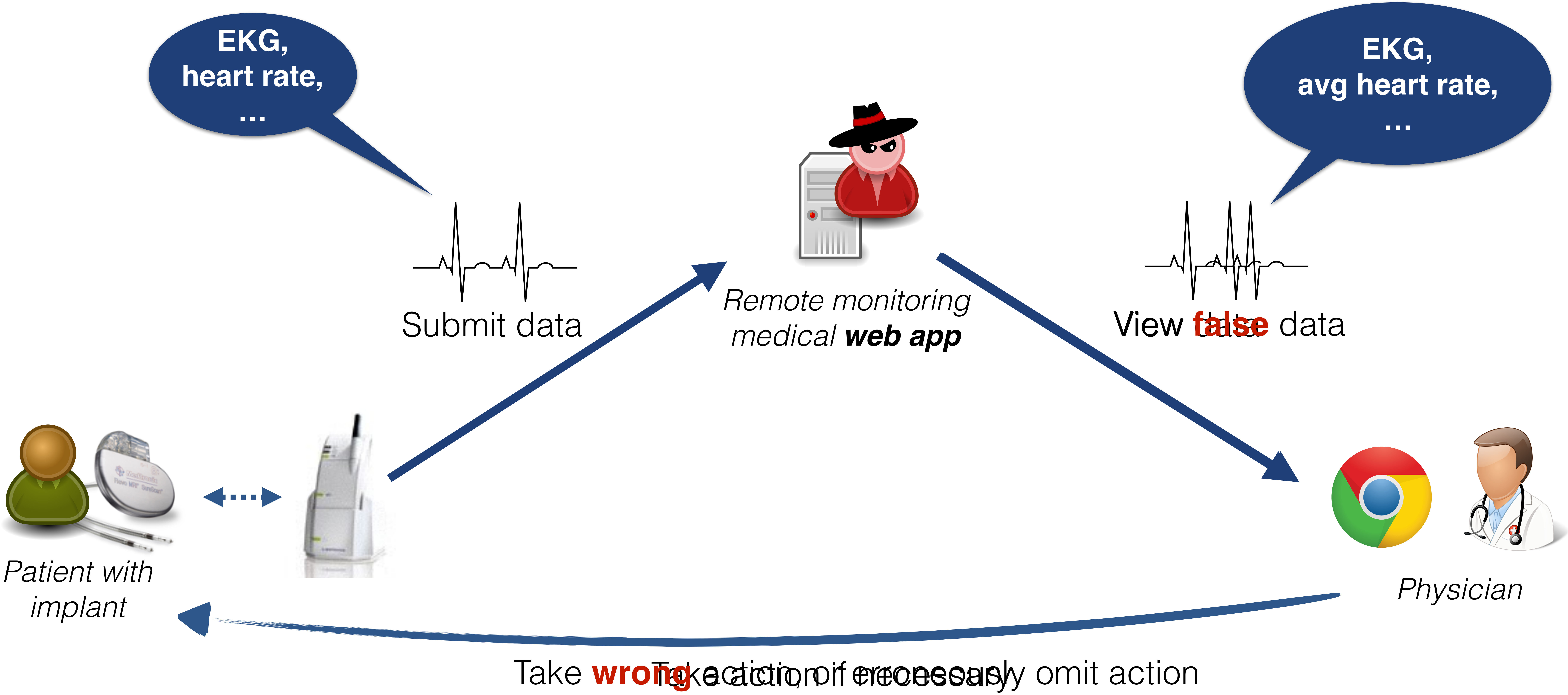
Verena:
End-to-End Integrity Protection for Web Applications

IEEE Security & Privacy 2016

Nikos Karapanos, Alexandros Filios, Raluca Ada Popa, Srdjan Capkun

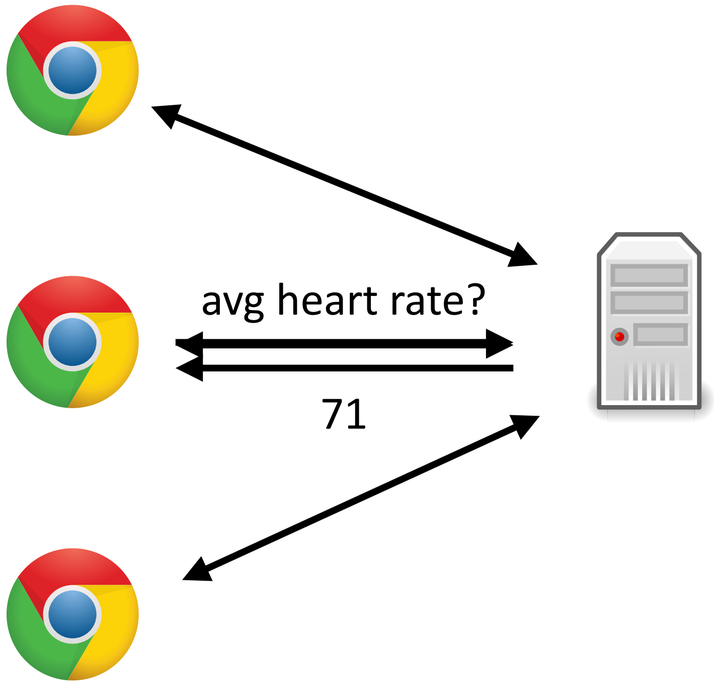
Information Integrity is Critical for Decision Making



Problem Definition

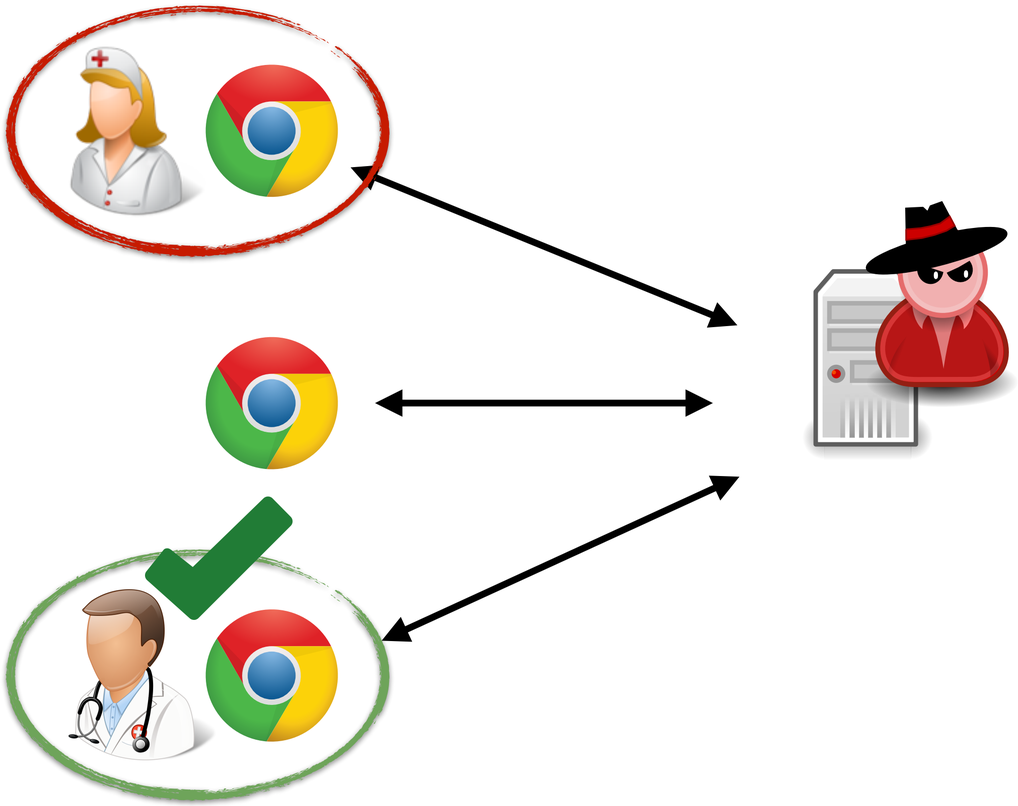
- How can we provide integrity guarantees in web applications?

- Example: Mean heart rate of a patient over a period of time
  - *Correctness*
  - *Completeness*
  - *Freshness*



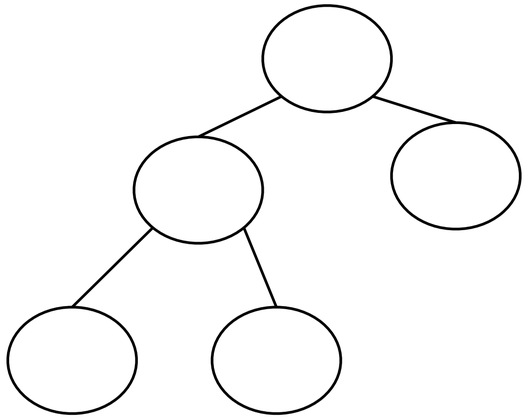
Threat Model

- Full server compromise (front-/back- end)
- Corrupted server responses
  - False (*correctness*)
  - Incomplete (*completeness*)
  - Stale (*freshness*)
- Clients are not fully trusted either



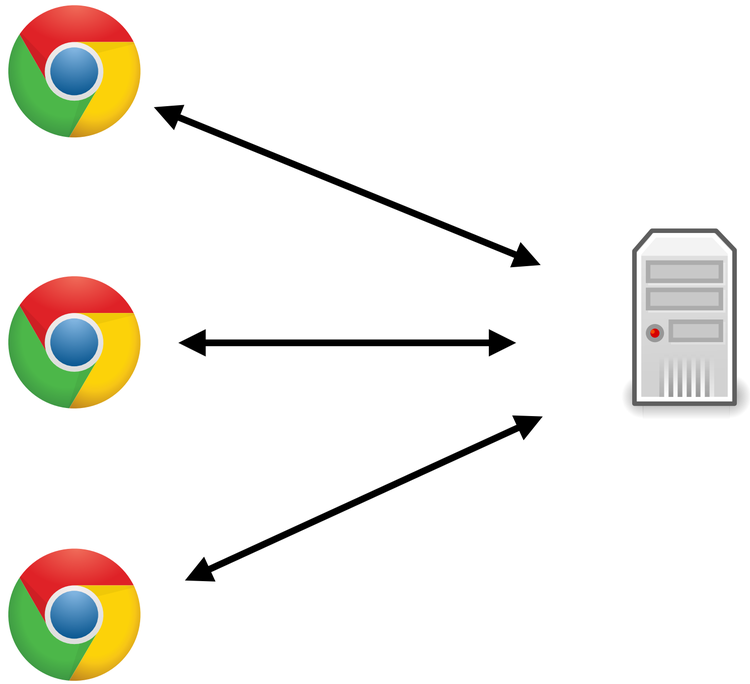
Related Work

- Filesystem integrity
  - SUNDR (OSDI'04),…
- Database integrity
  - IntegriDB (CCS'15),…
- Authenticated data structures
  - Balanced Merkle hash trees
  - Skip lists
  - …



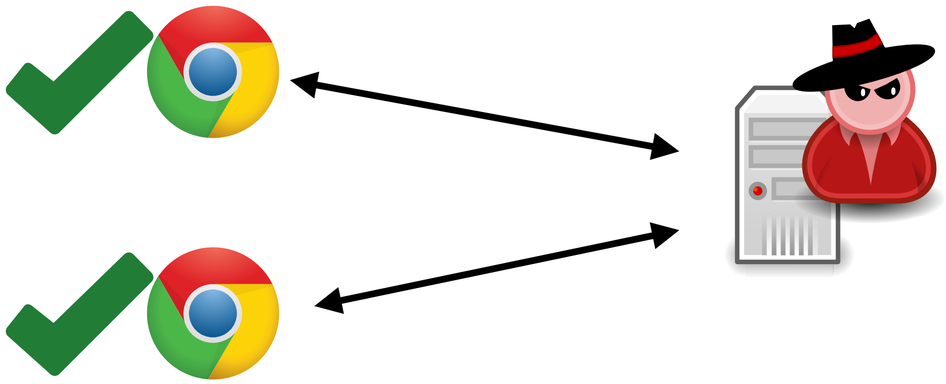
Challenges

- Multiple users in a dynamic environment
  - No single data owner
- Stateless clients, not always on
- How can the developer express the integrity policy?
  - *Don't change coding patterns*



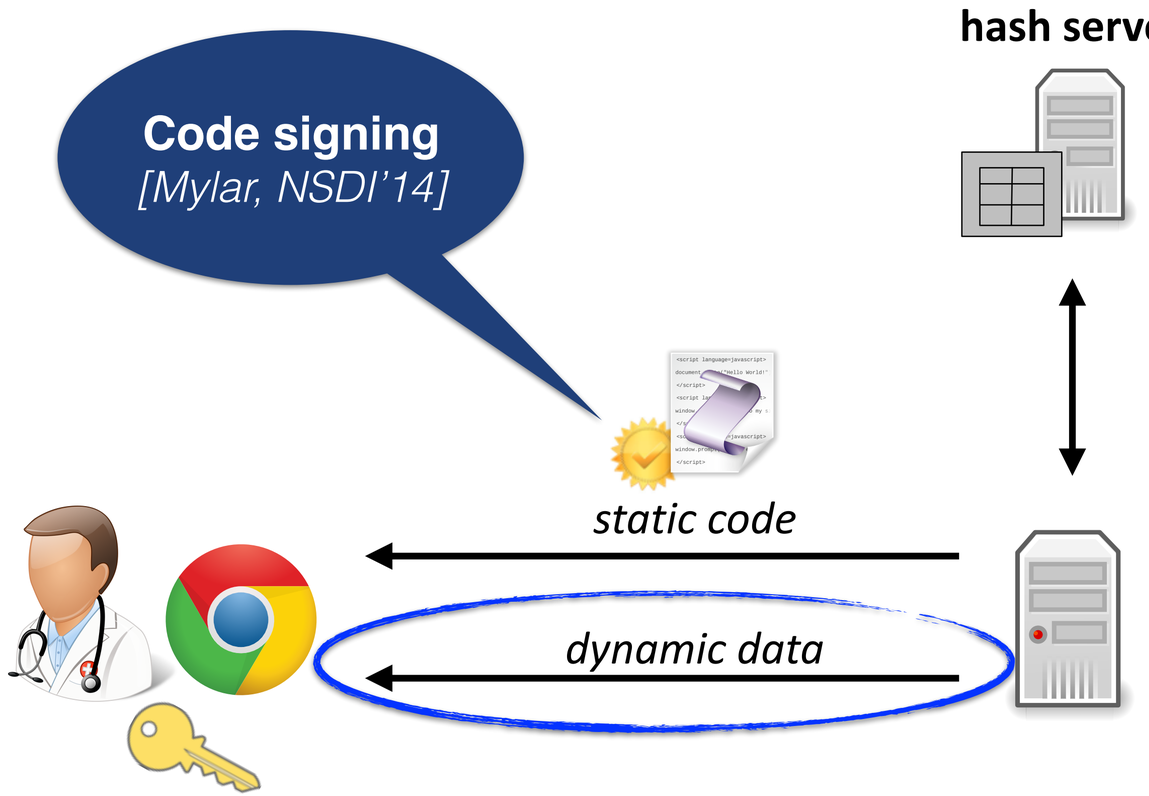
Verena

Framework for providing end-to-end integrity guarantees in web applications



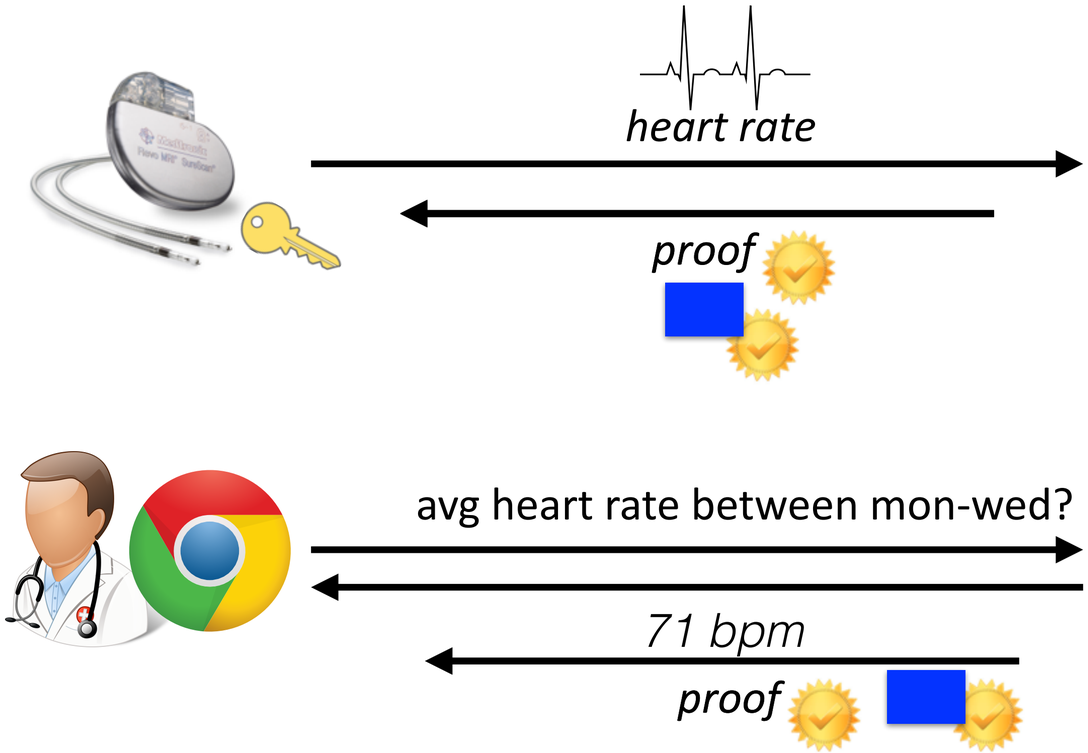
Verena Architecture: Setup

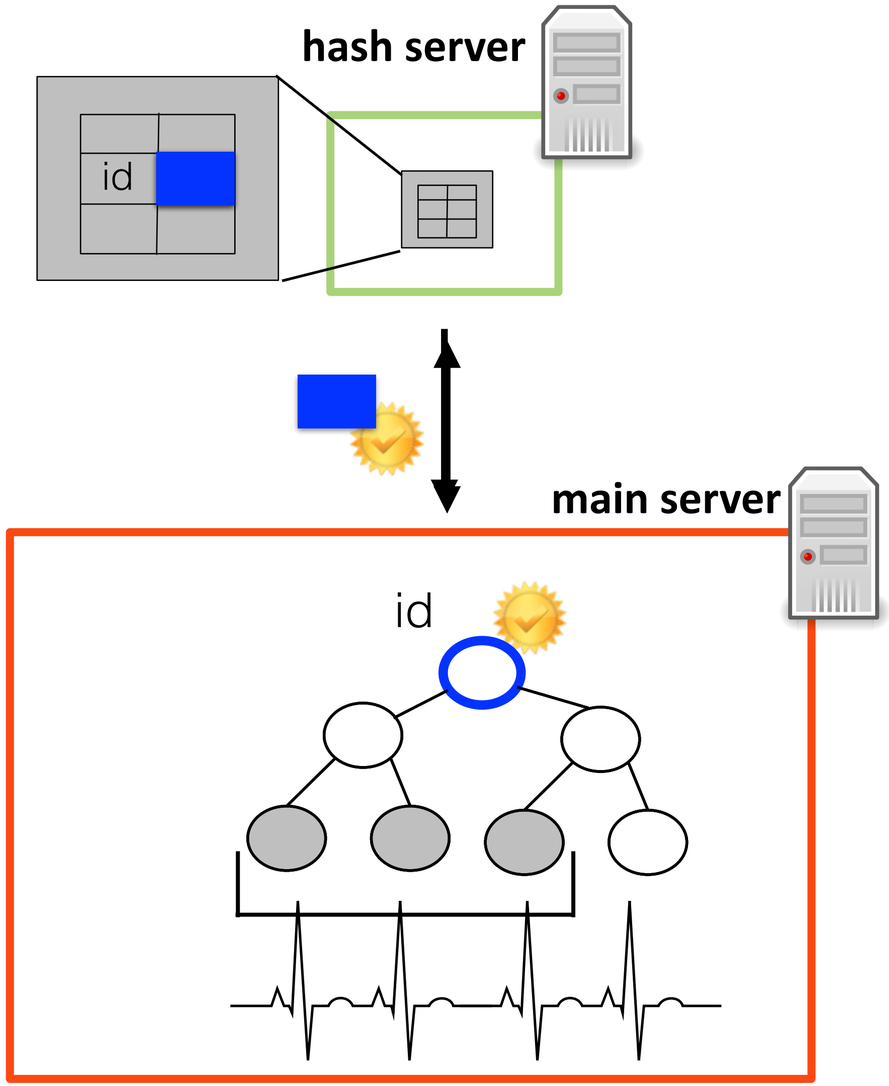
- Users
  - Key pair
  - Sign write operations
- Client-side web application
  - Code & data separation
  - Dynamic page rendering on the client
- Hash server
  - Ensure freshness
  - *Simple logic, narrow interface*



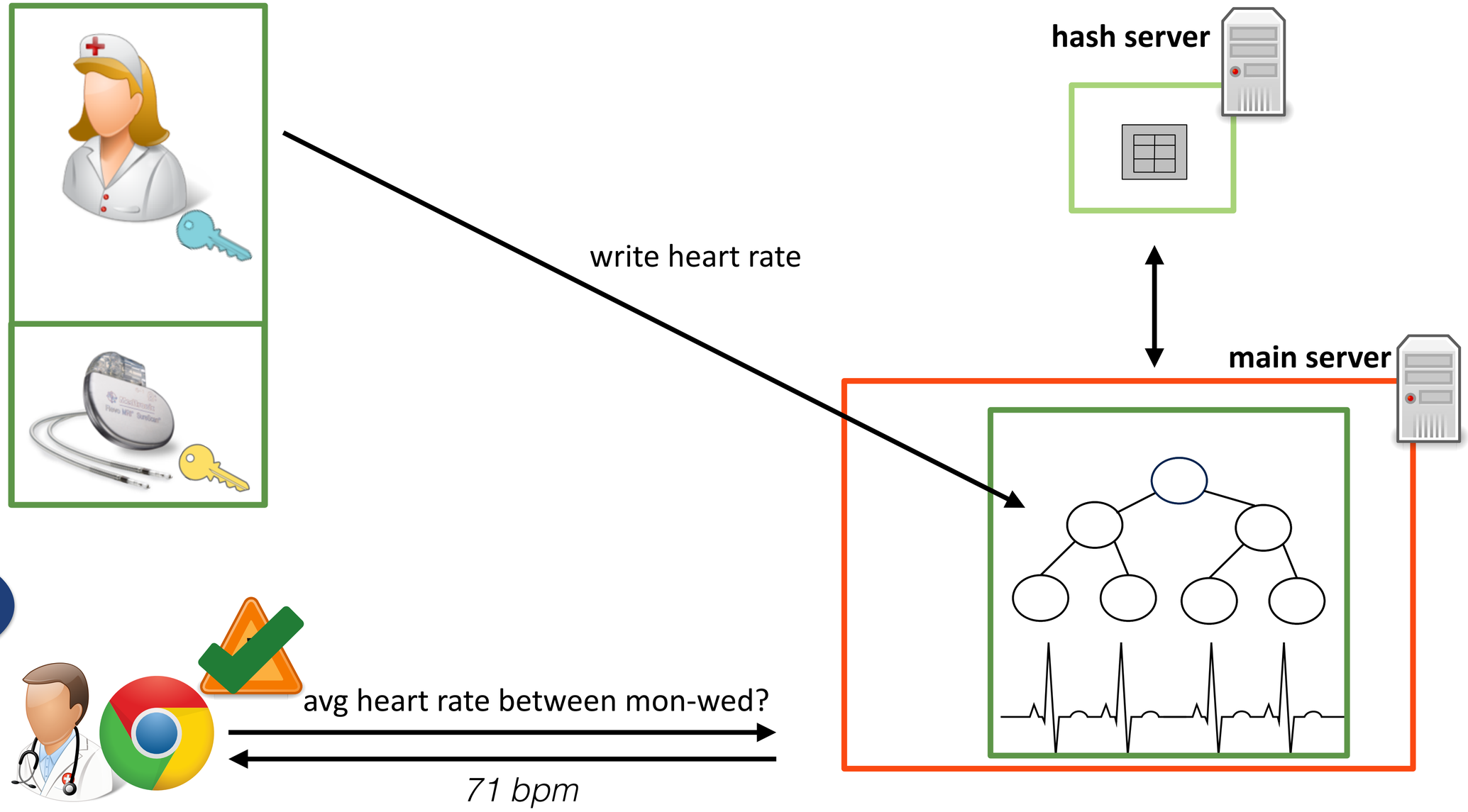
Verena Architecture

- Is the result **correct** and **complete**?
- Is the result **fresh**?
- Was the result affected by **authorized** users?





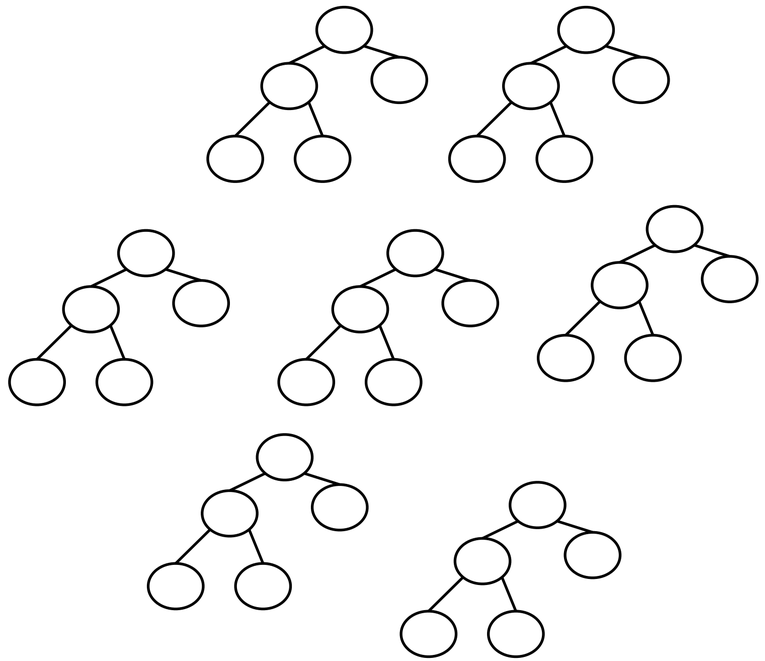
Trust Contexts



Verena API

- **Each query runs within a trust context**
  - Ability to run over multiple trust contexts and still ensure completeness
- API to manage trust contexts
- Annotate using Integrity Query Prototypes

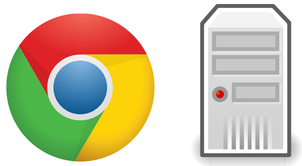
Queries



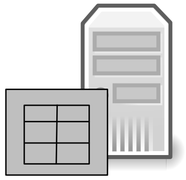
Implementation

- Platform of choice: Meteor framework (Node.js)

- Main server/client: Meteor package
  - Chrome Native Client for PK crypto in browser
  - **~5100 LOC**



- Hash server in Golang/OpenSSL
  - RocksDB key/value store
  - **~650 LOC**



Some Evaluation Results - Remote Monitoring Medical Application

- Page loading time for various views
  - Patient list (*~66ms*)
  - Patient for review (*~82ms*)
  - Patient profile (*~14ms*)
  - Patient EKG (*~23ms*)
  - Mean heart rate (*~13ms*)

VS

- Vanilla Meteor
  - An order of magnitude faster (*3-10ms*)

User experience is not affected

Verena provides end-to-end integrity protection to web applications

Under web server compromise

With acceptable overhead

Thank you for your attention!
Any Questions?

knikos@inf.ethz.ch

Some of the icons used in this presentation were taken and adapted from opensecurityarchitecture.org