Building a common and privacy-preserving front end for open-source clinical research platforms

_Presentation @ European i2b2 tranSMART AUG, Geneva, 01/11/2018_

Mickaël Misbach**, Ward Weistra**, Dr. Bo Gao**, Dr. Jean-Louis Raisaro*, Dr. Juan Troncoso-Pastoriza*, Prof. Jean-Pierre Hubaux*

*EPFL, #The Hyve
Motivation

Limitations:

- data fragmented in…
  1. location
  2. technical solution
- sensitive data not easily shared
Objectives

**Goals:** Extend Glowing Bear capabilities
- integrate additional clinical research platform: i2b2
- enable privacy-preserving cohort exploration: MedCo

**Why:** Enable scientists to access more data from a common interface by...
- expanding compatibility of data source backends
- accessing sensitive data that would otherwise be difficult to share

**How:** Add a layer of interoperability on top of Glowing Bear to support i2b2 and MedCo
Building Blocks
Common User Interface: Glowing Bear
Common Query Language: PIC-SURE API[1]

- PIC-SURE provides a common API to query any kind of data sources
- At the technical level, data semantic is not covered

PIC-SURE: Patient-centered Information Commons: Standardized Unification of Research Elements
IRCT: Inter-Resource Communication Tool
HMS-DBMI: Harvard Medical School - Department of Biomedical Informatics

i2b2

→ cohort exploration
→ database: star schema
→ preference from hospitals

tranSMART

→ i2b2 features and...
→ advanced cohort exploration features
→ advanced data export
→ preference from pharma
→ study-based

[1]: Shawn N Murphy et al. “Serving the enterprise and beyond with informatics for integrating biology and the bedside (i2b2)”. In: Journal of the American Medical Informatics Association 17.2 (2010), pp. 124–130.
MedCo[1]

Distributed cohort exploration
Secure storage outsourcing
Trust decentralization

End-to-end data protection
Unlinkability
Differential privacy

⇒ MedCo provides technical means to share data that would otherwise not be shared

MedCo

Initialization phase

\[ (K_1, k_1) + (K_2, k_2) + (K_3, k_3) + (K_4, k_4) = (K, K) \]

Site \( (S_1) \)

Query Processing

Site \( (S_2) \)

Site \( (S_3) \)

Site \( (S_4) \)

Query

# of Patients

Investigator \( (I) \)

i2b2 Web Client

Building a common and privacy-preserving front end for open-source clinical research platforms,
Mickaël Misbach, European i2b2 tranSMART AUG, Geneva, 01/11/18
Clinical site’s zoomed-in view

EHR System

Clinical Data Warehouse

MedCo ETL

i2b2 encrypted database

Pic-Sure

I2b2 hive

Encrypted Data Exit

Clinical (non-connected) zone

Research (connected) zone

ETL: Extraction Transformation (Encryption) and Loading tool

Clear data

Encrypted data

Building a common and privacy-preserving front end for open-source clinical research platforms, Mickaël Misbach, European i2b2 tranSMART AUG, Geneva, 01/11/18
System Overview
System Overview

HTTP REST API
- Client Side
- Server Side

API call

JWKS retrieval

Glowing Bear
- Interoperability Module
- Crypto Module

Keycloak

transSMART API v2

transSMART 17.1

i2b2

MedCo Node 1

MedCo Node 2

...

MedCo Node n

IRCT (PIC-SURE)

PIC-SURE API

OpenID Connect API

Interoperability Layer

Clinical Research Systems

Front End

Glowing Bear

Crypto Module

Interoperability Module

API call

JWKS retrieval

transSMART API v2

i2b2 API

i2b2 API

i2b2 API
Demo
Conclusion

- offer a modern front end that allows cohort exploration: Glowing Bear ✔
- compatible with tranSMART (v17.1) and i2b2 ✔
- compatible with MedCo ✔
- be extensible for future support of additional platforms ✔
- technical considerations:
  1. easy to deploy ✔
  2. not degrade user experience in existing systems ✔
  3. enforce secure authentication ✔
  4. open-source ✔
  5. practical runtime ✔

Building a common and privacy-preserving front end for open-source clinical research platforms, Mickaël Misbach, European i2b2 tranSMART AUG, Geneva, 01/11/18
Future Work

- Authentication: using a distributed ledger, to avoid single point of failure (Keycloak) (in progress)
- Support additional systems through the PIC-SURE API (in progress: HAIL[1], framework for exploration and analysis of genomic data)
- More features in UI for i2b2 and MedCo e.g. data export, analysis, query saving, etc.
- PIC-SURE 2.0 upgrade

[1]: Hail, an open-source, scalable framework for exploring and analyzing genomic data: https://github.com/hail-is/hail
Context

Project is a collaboration between EPFL & The Hyve
→ The Hyve: wanted Glowing Bear to support i2b2
→ EPFL: wanted new front end for MedCo
→ Within the framework of the Swiss DPPH (Data Protection in Personalized Health)

talk with chair guy for presentation
part of work being sponsor by dpph
User Interface before: i2b2 webclient
Website: medco.epfl.ch

Roadmap
→ MedCo with PIC-SURE and Glowing Bear: Nov. 2018
→ MedCo with MedChains (blockchain-based authentication and access control): April 2019
→ Skype-like pull model: June 2019
OpenID Connect[1] / Keycloak

OpenID Connect
➔ authentication and authorization protocol
➔ token-based
➔ allow any kind of services to delegate identity and access management

Keycloak
➔ identity provider
➔ implements OpenID Connect server

Why?
➔ many different components running on different systems, with different authentication mechanisms
➔ need to have common authentication

[1]: Nat Sakimura et al. “OpenID Connect Core 1.0 incorporating errata set 1”. In: The OpenID Foundation, specification (2014).
System Design Overview
Glowing Bear[1] 🐻

➔ User interface for cohort exploration
➔ And more advanced analytics
➔ Open-source web application
➔ Originally for tranSMART v17.1 only

[1]: Glowing Bear, a modern cohort selector for i2b2 tranSMART: https://glowingbear.app/
PIC-SURE API

- Query terms of data source exposed through a tree
- Each data source declares its query format
- Data source has freedom of implementing anything as long as it fits the interface
- API has a SQL-like format:

```json
"where": [{
  "field": {
    "pui": "/resource/study/Age/",
    "dataType": "INTEGER"
  },
  "predicate": "CONSTRAIN_VALUE",
  "fields": {
    "OPERATOR": "GT",
    "CONSTRAINT": "20"
  }
}],
```

*where part: constraints on data*

*field: path and type of query term queried (obtained from tree)*

*predicate used on query term*

*fields: additional input to predicate*
IRCT

- IRCT is the official implementation of the PIC-SURE API
- 4 components:
  - IRCT-CL (REST service)
  - IRCT-API (core library)
  - IRCT-EXT (external hooks library)
  - IRCT-RI (data sources connectors)

IRCT: Inter-Resource Communication Tool
CL: Communication Layer
API: Application Programming Interface
RI: Resource Interface
EXT: EXTension
Detailed Workflows
Detailed Query Workflow

1. User Login
2. GB Initialization
3. Query Construction
4. Query Submission
5. Query Translation
6. Query Processing
7. Result Storage
8. Result Display
OpenID Connect: Stateless Authentication

1. Get JWT
2. HTTP Request, with JWT
3. Get public key

Legend:
- JWT: JSON Web Token
- Cached Request
- HTTP Request
- Server Side
- Client Side
OpenID Connect: JSON Web Token

JWT format

- 3 distinct base64-encoded values
  - JSON header: metadata
    ```json
    {
      "alg": "RS256",
      "typ": "JWT",
      "kid": "eTFrdyrNxXLNHI7p0Ywybc7z1SBHTEcqWcMTybtvQY"
    }
    ```
  - JSON payload: identity, authorizations, validity, ...
    ```json
    {
      "exp": 1523454086,
      "iat": 1523453186,
      "iss": "http://localhost:8081/auth/realms/master",
      "aud": "glowing-bear",
      "nonce": "N0.28573339803406971523453198656",
      "resource_access": {
        "account": {
          "roles": [
            "role1",
            "role2"
          ]
        },
        "preferred_username": "test",
        "email": "test@test.com"
      }
    }
    ```
  - Binary signature
Implementation
Implementation of OpenID Connect user authentication in Glowing Bear
Glowing Bear: Interoperability Module

- Implementation of interoperability module in Glowing Bear
- Implementation of PIC-SURE API client side
Glowing Bear: Crypto Module

Integration of crypto module in Glowing Bear
Glowing Bear: User Interface

- **Modification** of Glowing Bear to enable or disable UI elements according to data source
IRCT: OpenID Connect

Modification of IRCT to support a new type of JWT signature
IRCT: Cross-Origin Resource Sharing (CORS)

Implementation of Cross-Origin Resource Sharing support in IRCT, a web security standard for HTTP requests in browsers
IRCT: i2b2 Data Source

Modifications of the i2b2 data source support in IRCT, to support OpenID Connect authentication and add features
IRCT: MedCo Data Source

- **Implementation** of the MedCo data source support in IRCT
i2b2: OpenID Connect

Implementation in i2b2 of OpenID Connect authentication (also used by MedCo)
MedCo: IRCT Querying

Modification of MedCo to support queries from IRCT, this includes removal of SHRINE support
Conclusion
Backups
Query Workflow: MedCo

1. User Login
2. GB Initialization
3. Query Construction
4. Query Submission
5. Query Translation
6. Query Processing
7. Result Storage
8. Result Display
Query Workflow: MedCo

Step 2: Glowing Bear Initialization
➔ Public key of the collective authority loaded (collective authority = all the MedCo nodes):
  \( Pk_c \)
➔ Pair of public/private key of user randomly generated
  \( Pk_u / pk_u \)

Step 3: Query Construction
➔ In the tree, each query term has an a corresponding integer
  \( q_v \)
➔ It is encrypted before submission with the public key of the collective authority
  \( ENC_{PkC}[q_v] \)

Step 5: Query Translation
➔ Query in PIC-SURE API format translated to i2b2 API format
➔ Broadcasted to all the MedCo nodes at the same time
Query Workflow: MedCo

Step 6: Query Processing (at each node, using distributed protocols involving all nodes)

A. Encrypted query terms are “tagged”: the encryption is switched from probabilistic to deterministic (DDT = Distributed Deterministic Tagging)

\[ \text{ENC}_{\text{Pkc}}[q_v] \rightarrow \text{DDT}_{\text{si}}[q_v] \]

B. The query is submitted to normal i2b2 (in the database, the \( \text{DDT}_{\text{si}}[q_v] \) are stored)

C. i2b2 answers with a patient set, but it contains dummy patients
Query Workflow: MedCo

Step 6: Query Processing (distributed protocols involving all the MedCo nodes)

D. Each patient has a “dummy flag”: an encrypted 0 or 1
   → we fetch the flags of patients from the set
   \[ ENC_{\text{PkC}}^{f_j} \]

E. The sum of the flags is our real count, we compute it homomorphically:
   \[ ENC_{\text{PkC}}^{R_i} \]

F. The result is encrypted with the key of the collective authority, the corresponding private key does not exist
   → we change the key of the encryption to the one of the user
   \[ ENC_{\text{PkU}}^{R_i} \]
Query Workflow: MedCo

Step 8: Result Display

- The encrypted results are fetched by Glowing Bear
- They are decrypted using the private key of the user $R_i$