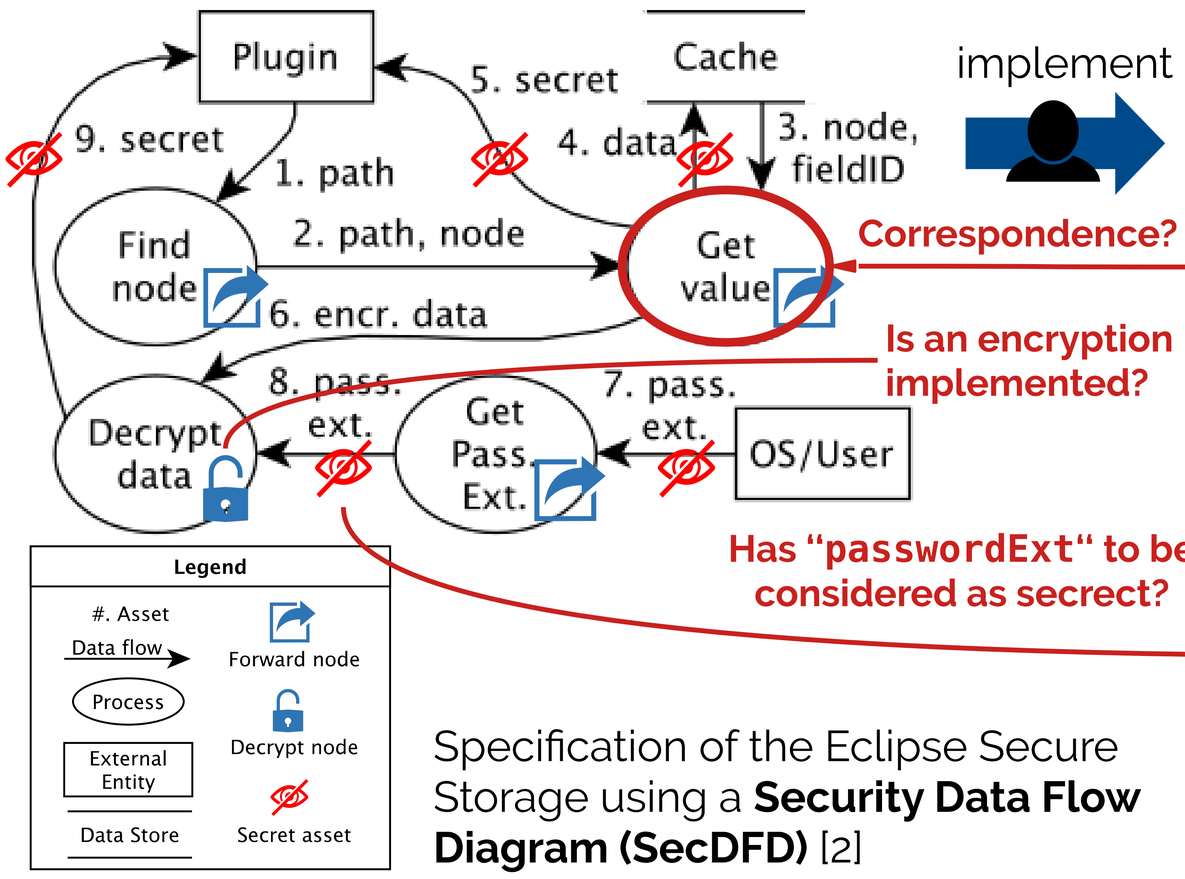
Secure Data-Flow Compliance Checks between Models and Code based on Automated Mappings

Sven Peldszus[1], Katja Tuma[2], Daniel Strüber[2], Jan Jürjens[1,3], Riccardo Scandariato[2]

1) University of Koblenz-Landau 2) University of Gothenburg and Chalmers University of Technology 3) Fraunhofer ISST

CONTEXT

According to the principle of **security by design**, the system's assets and threats have to be defined in the earliest phases of development.



Specification of the Eclipse Secure Storage using a **Security Data Flow Diagram (SecDFD)** [2]

PROBLEM DESCRIPTION

> Does the implementation comply to the model?

> Do the designed security properties hold?

→ For checking the compliance we need a mapping between models and code

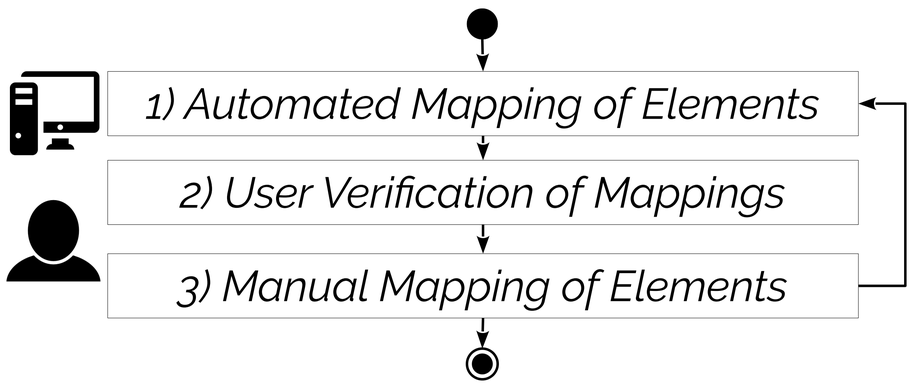
```
 30 public class SecurePreferences {
    ...
251   public String get(String key, String def,
        SecurePreferencesContainer container) ... {
      ...
255     String encryptedValue = internalGet(key);
256     if (encryptedValue == null)
257       return null;
258
259     CryptoData data = new CryptoData(encryptedValue);
260     String moduleID = data.getModuleID();
        ...
267     PasswordExt passwordExt = getRoot()
          .getPassword(moduleID, container, false);
        ...
277   }
      ...
508 }
```

SEMI-AUTOMATED APPROACH

Mappings between SecDFDs and program models for:

> Compliance checks
- > Convergence
- > Divergence
- > Absence

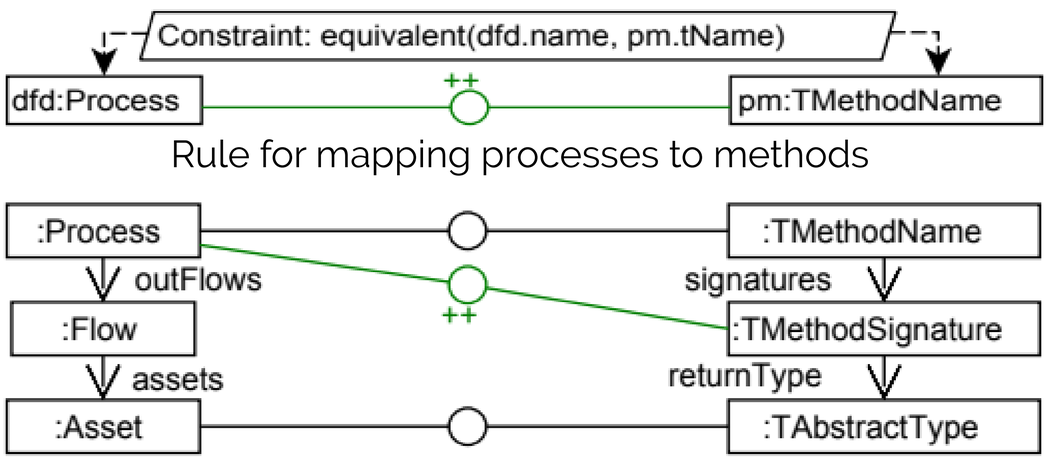
> Transfer of security labels
- > Security metrics
- > Secure data flow analyses
- > Runtime-monitoring



Automated Mapping of Elements

> Based on **GRaViTY** program model [3]
- > Abstracts details from statement level
- > Detailed type and method information

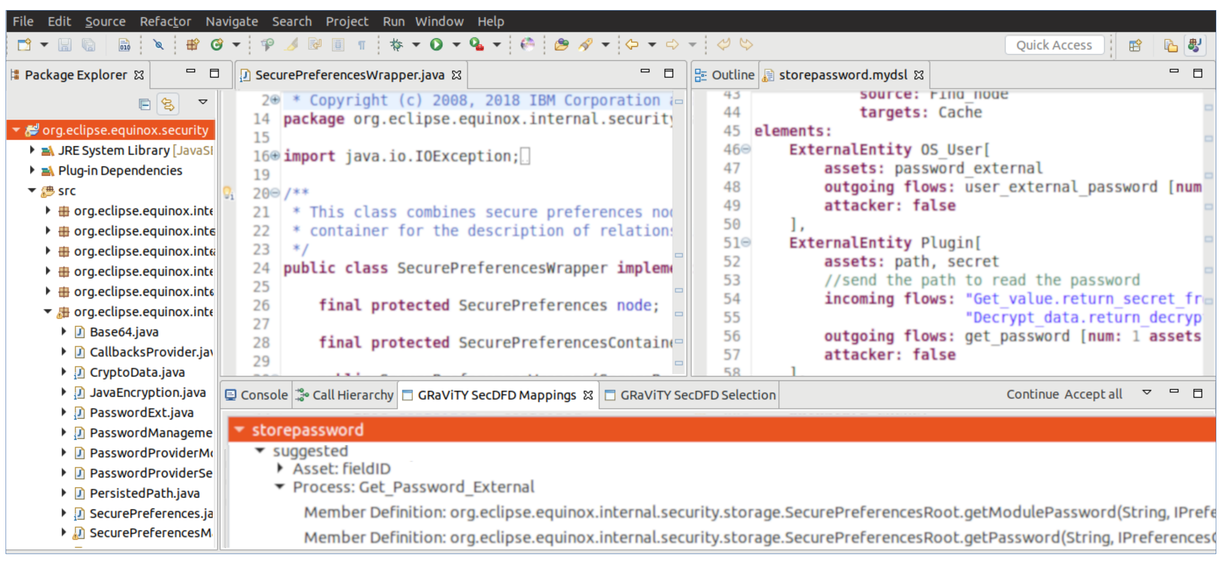
> Multiple mapping rules and heuristics



User Verification of Mappings and *Manual Mapping of Elements* integrated into the Eclipse IDE

> Accept Mapping → Taken as valid

> Reject Mapping → Never shown again



VALIDATION

Executed Mapping on 5 realistic open source projects

> 1st automated iteration: mean precision 50.5%; mean recall 69.8%

> Last automated iteration: mean precision 87.2%; mean recall 92.0%

> **75% of the correct mappings have been suggested**

GET THE TOOL!

secdfd.gravity-tool.org

LITERATURE

[1] S. Peldszus, K. Tuma, D. Strüber, J. Jürjens, R. Scandariato, "Secure Data-Flow Compliance Checks between Models and Code based on Automated Mappings," in MODELS, 2019

[2] K. Tuma, M. Balliu, and R. Scandariato, "Flaws in Flows: Unveiling Design Flaws via Information Flow Analysis," in ICSA, 2019

[3] S. Peldszus, G. Kulcsár, M. Lochau, and S. Schulze, "Incremental Co-Evolution of Java Programs based on Bidirectional Graph Transformation," in PPPJ, 2015