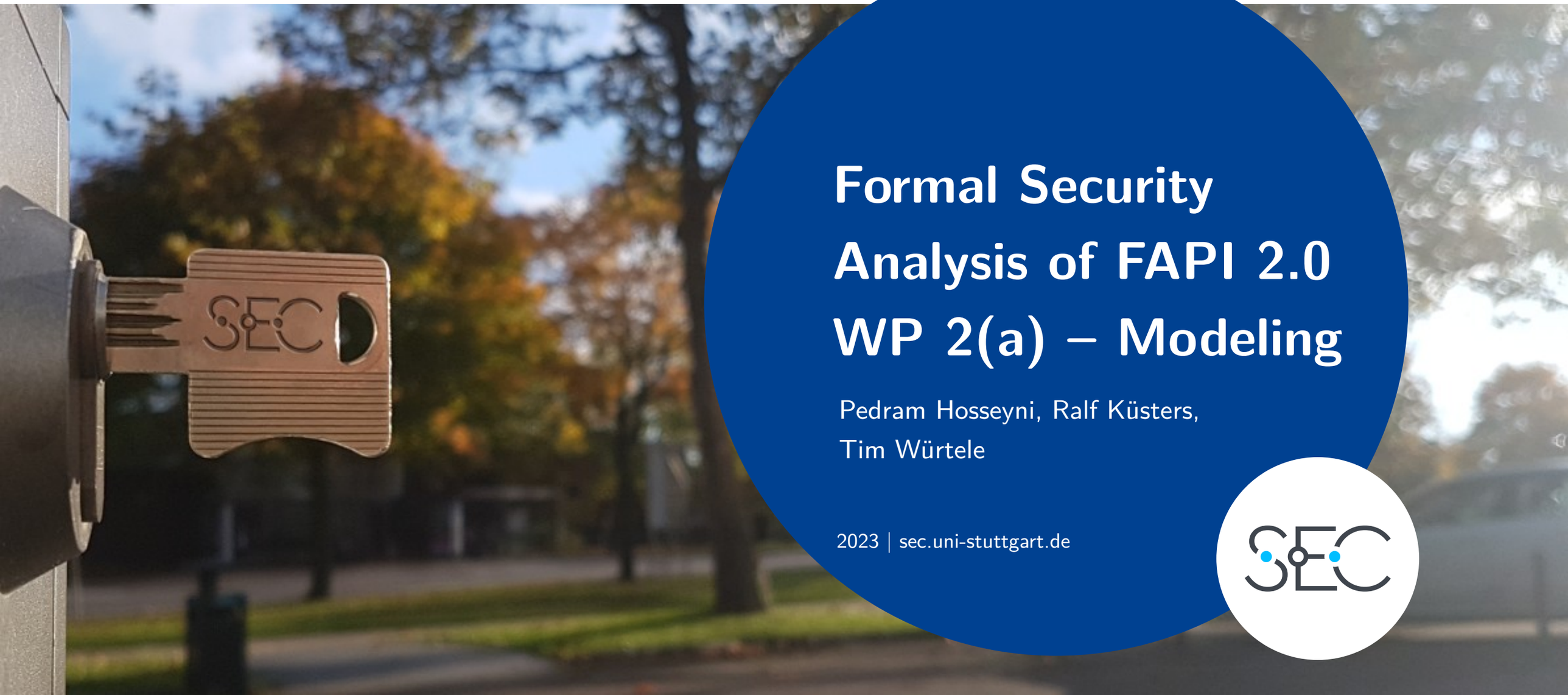
University of Stuttgart
Institute of
Information Security



Formal Security Analysis of FAPI 2.0 WP 2(a) – Modeling

Pedram Hosseyni, Ralf Küsters,
Tim Würtele

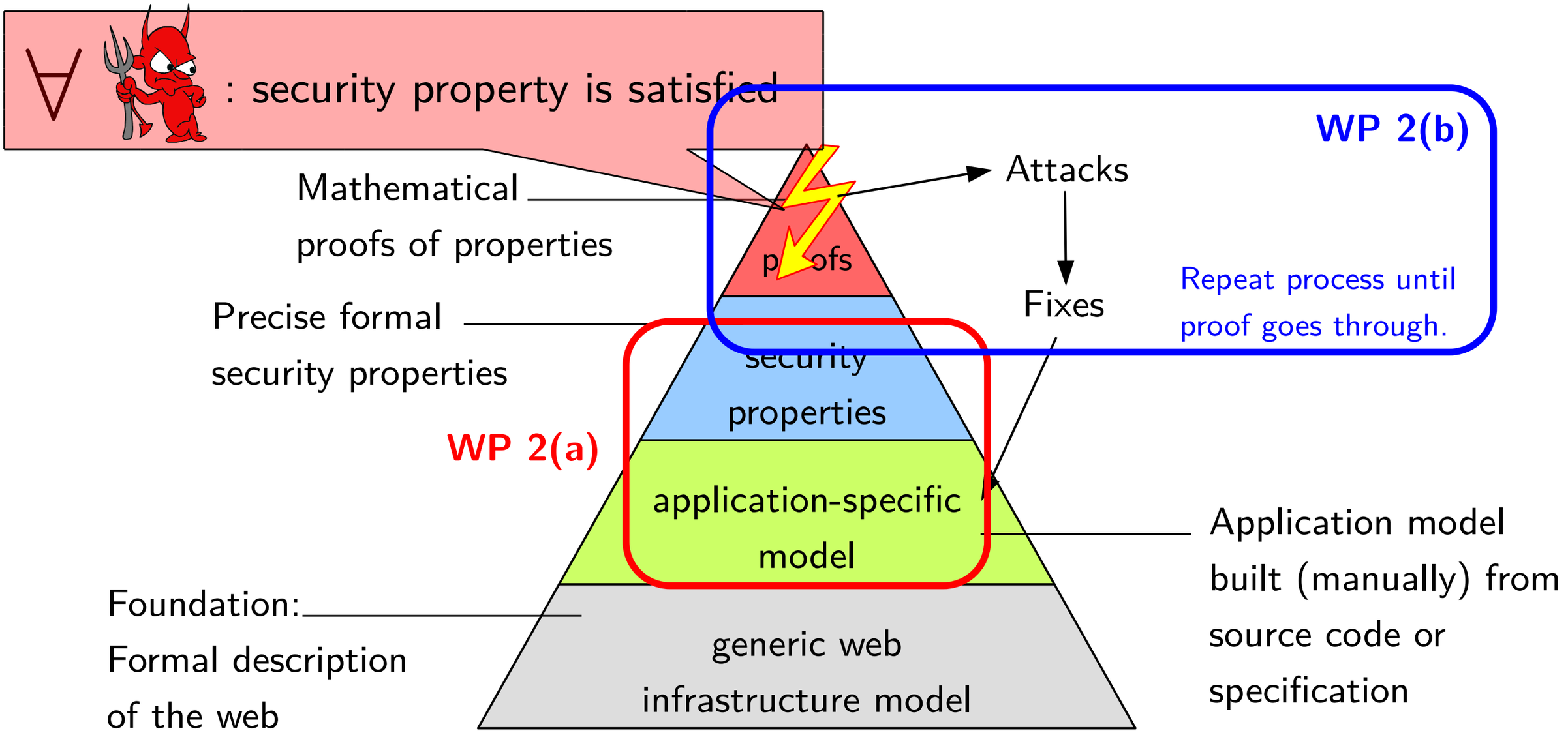
2023 | sec.uni-stuttgart.de

Outline

- The Web Infrastructure Model
- Modeling
- Notes on Specifications
- Security Properties

The Web Infrastructure Model (WIM)

The Web Infrastructure Model *WIM*



Modeling

Modeled Components

- FAPI 2.0 Security Profile
  - Everything else in context of FAPI 2.0 Security Profile
- Dynamic Client Registration & Management
- FAPI-CIBA
- FAPI 2.0 Message Signing

Noteworthy Modeling Details

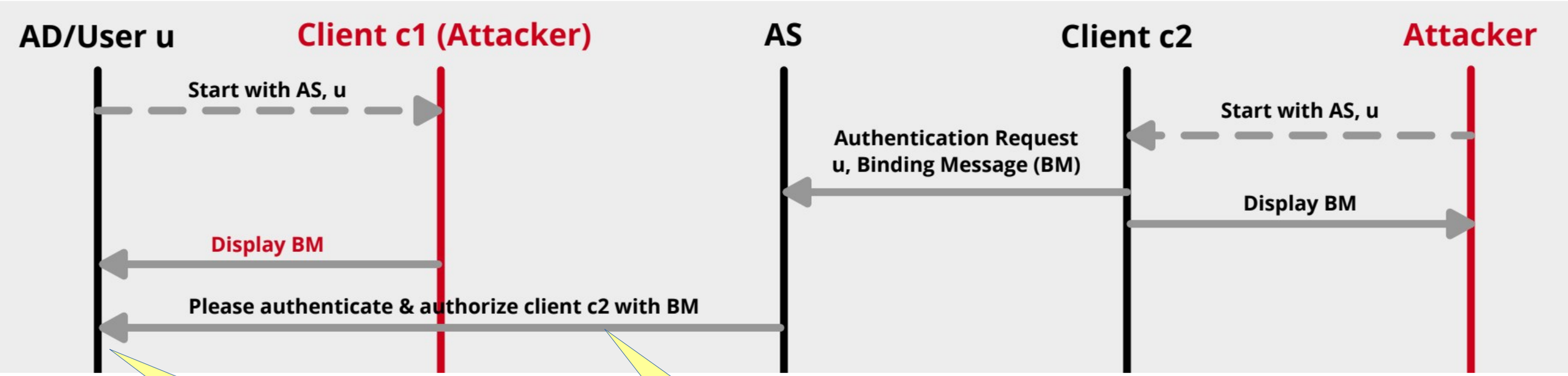
- Dynamic Client Registration
  - No registration access token
- Dynamic Client Management
  - Attacker-timed updates of client key material/deletion of client registration
  - Immediate key rollover
- FAPI-CIBA
  - Attacker-controlled flow initiation (incl. user selection)
  - Consumption device subsumed by client
- FAPI 2.0 Message Signing
  - Interleaving flows with & without signing possible

Notes on Specifications

Dynamic Client Management

- Semantics of client key rollover
  - What about ongoing grants?
  - Issued tokens?
  - Token introspection?
  - Compromised keys?

FAPI-CIBA



=> AS should provide user with client information in addition to binding message

Security Properties

Security Goals

- FAPI 2.0 Attacker Model specifies three security goals
  - Authorization: “no attacker can access resources belonging to a user”
  - Authentication: “no attacker is able to log in at a client under the identity of a user”
  - Session Integrity:
    - “no attacker is able to force a user to use resources of the attacker”
    - “no attacker is able to force a user to be logged in under the identity of the attacker”

Same security goals for FAPI-CIBA and when using Dynamic Client Registration and Management.

FAPI 2.0 Attacker Model: https://openid.bitbucket.io/fapi/fapi-2_0-attacker-model.html

Non-Repudiation Requirements

- In addition to security goals, FAPI 2.0 Message Signing also specifies non-repudiation requirements for:
  - Pushed Authorization Requests
  - Authorization Responses
  - Introspection Responses
  - Resource Requests/Responses

Not for regular Authorization Request (FAPI 2.0 mandates PAR)

Either both are signed, or none of them (see Subsection "Profile")

- General idea of formalized non-repudiation properties:

  There is a protocol participant that used a signing key to sign a certain message

FAPI 2.0 Message Signing: https://bitbucket.org/openid/fapi/src/575-httpsig/fapi-2_0-message-signing.md

Non-Repudiation: Example (Simplified)

AS required a signed PAR

Non-repudiation for signed pushed authorization requests (Definition 21, Page 44):

Let $\mathcal{FAPI}$ be a FAPI web system with a network attacker. We say that $\mathcal{FAPI}$ is secure w.r.t. *non-repudiation for signed authorization requests* iff for every run ρ of $\mathcal{FAPI}$, every configuration (S^n, E^n, N^n) in ρ, every process as $\in$ AS that is honest in S^n, every request uri requestUri, we have that if S^n(as).authorizationRequests[requestUri][signed_par] $\equiv \top$, then there exists a processing step $Q = (S, E, N) \rightarrow (S', E', N')$ with (S, E, N) prior to (S^n, E^n, N^n) in ρ in which as processes the input event e_{in}, such that requestUri $\notin$ S(as).authorizationRequests and requestUri $\in$ S'(as).authorizationRequests. In addition, e_{in} = `<x,y,m>` contains a message m of the form `<HTTPReq,·,POST,selectedAS,/par,·,<>,body>`, where body is of the form `sig(par, signKey)`. Furthermore, we have that:

AS received a PAR with a signed message

- $\exists$c $\in$ C such that all of the following hold
  1) c is honest in S^n
  2) S^n(c).asAccounts[selectedAS][sign_key] $\equiv$ signKey
  3) there exists a processing step Q' prior to Q in ρ, in which c outputs `<x',y',m>` and executes Line 60 of Algorithm 8 with requestData $\equiv$ par.

The client which is registered at AS with the signing key previously created the PAR