draft-meyerzuselhausen-oauth-iss-auth-resp

Karsten Meyer zu Selhausen, Daniel Fett

OAuth Interim Meeting, Dec. 7 2020

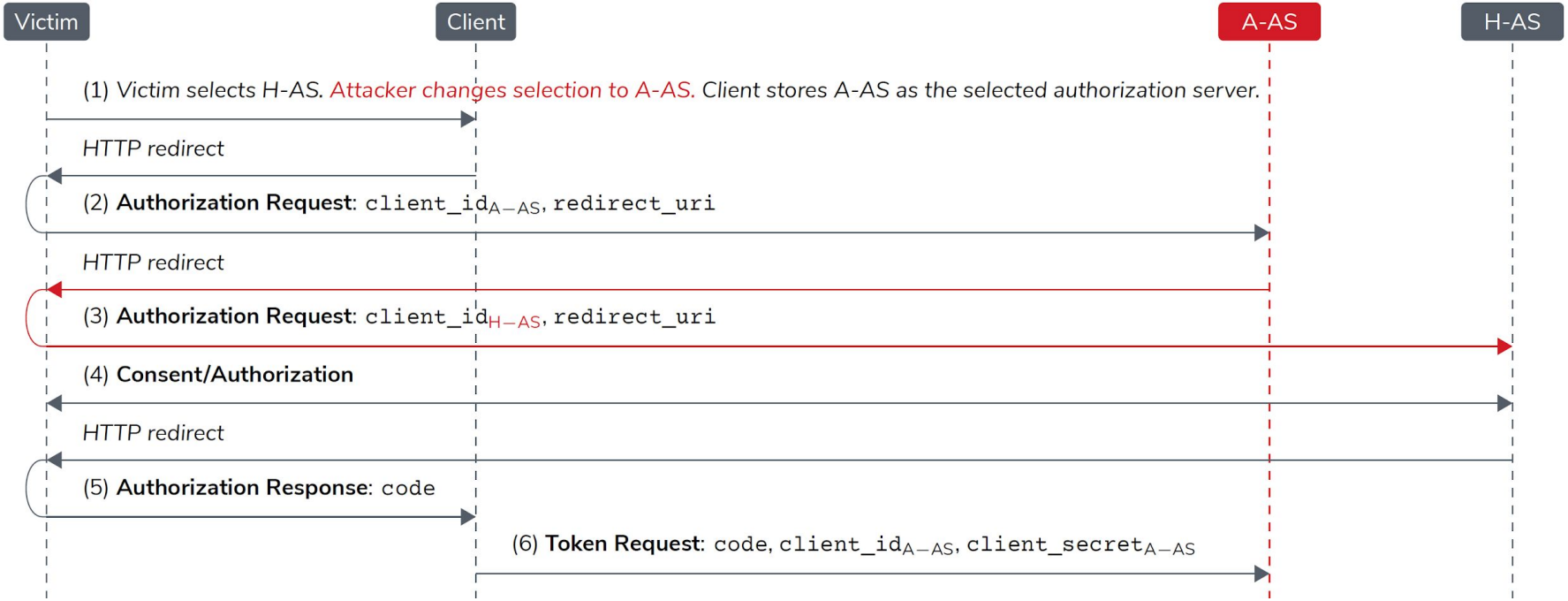
What are Mix-Up Attacks?

Mix-Up Attack Overview

- Goal: Steal authorization code or access token
- Idea: Trick client to send credentials to AS controlled by an attacker (A-AS) instead of honest AS (H-AS)
- Precondition: Client supports multiple AS, one controlled by an attacker
  - Attacker registers client at his AS using dynamic client registration
  - Attacker compromises an AS
- Different variants with additional preconditions
  - Possible for code and implicit grant
  - OIDC variant

Mix-Up Attack Variant

- Precondition: Attacker can manipulate the first request



How to Defend Against Mix-Up?

First Discussions

- Confidential Clients?
- PKCE?
- Per-AS Redirect URIs?
- iss-like Parameter?

Since then:

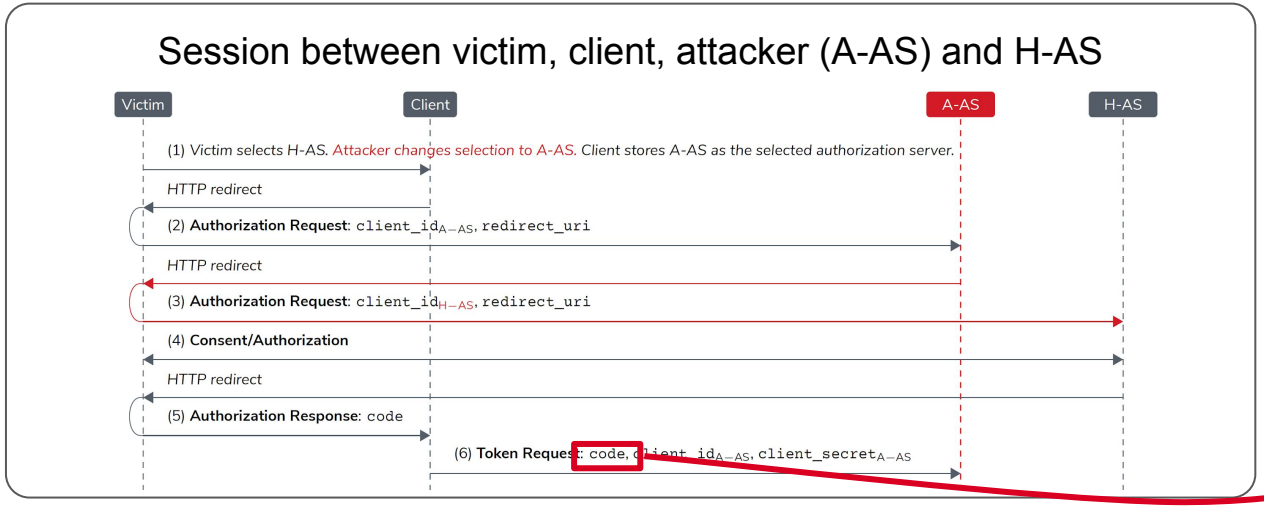
- Gathered practical experiences
- Refined security and threat considerations

0th OAuth Security Workshop, Darmstadt, 2015

Are Confidential Clients Safe?

No: Attacker can **inject stolen code into authorization response** in another session (under his control) with the client and H-AS. (Code Injection Attack)

Client will redeem the stolen code with credentials and give attacker access to victim's protected resources.

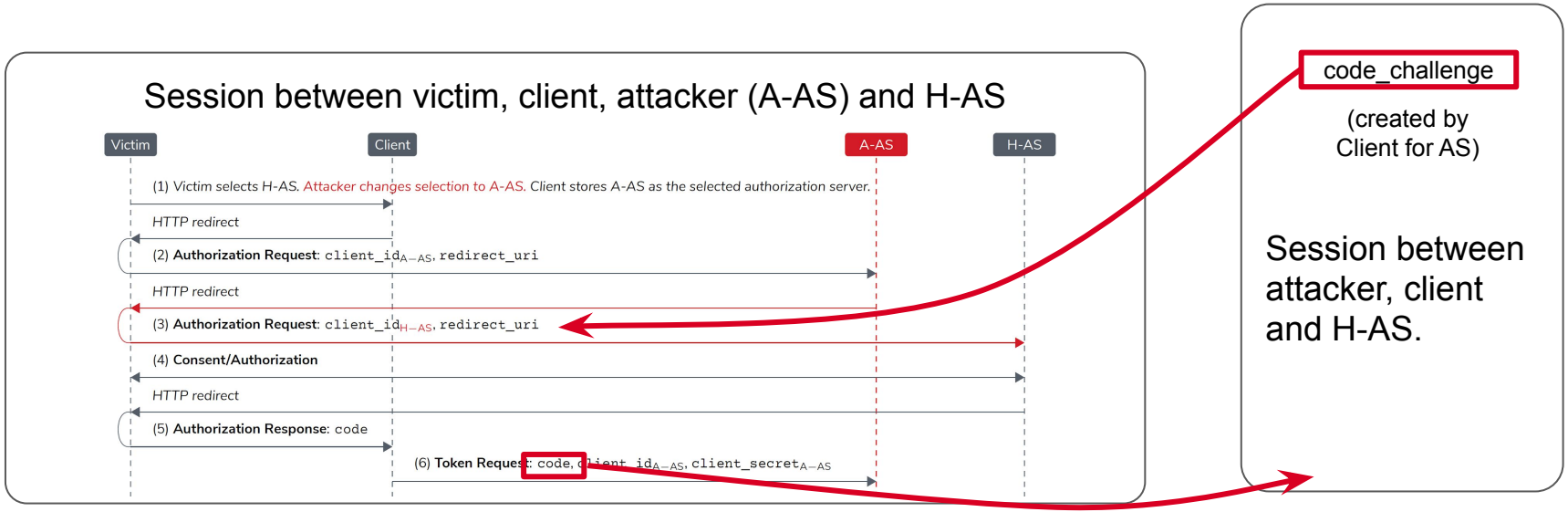


Session between attacker, client and H-AS.

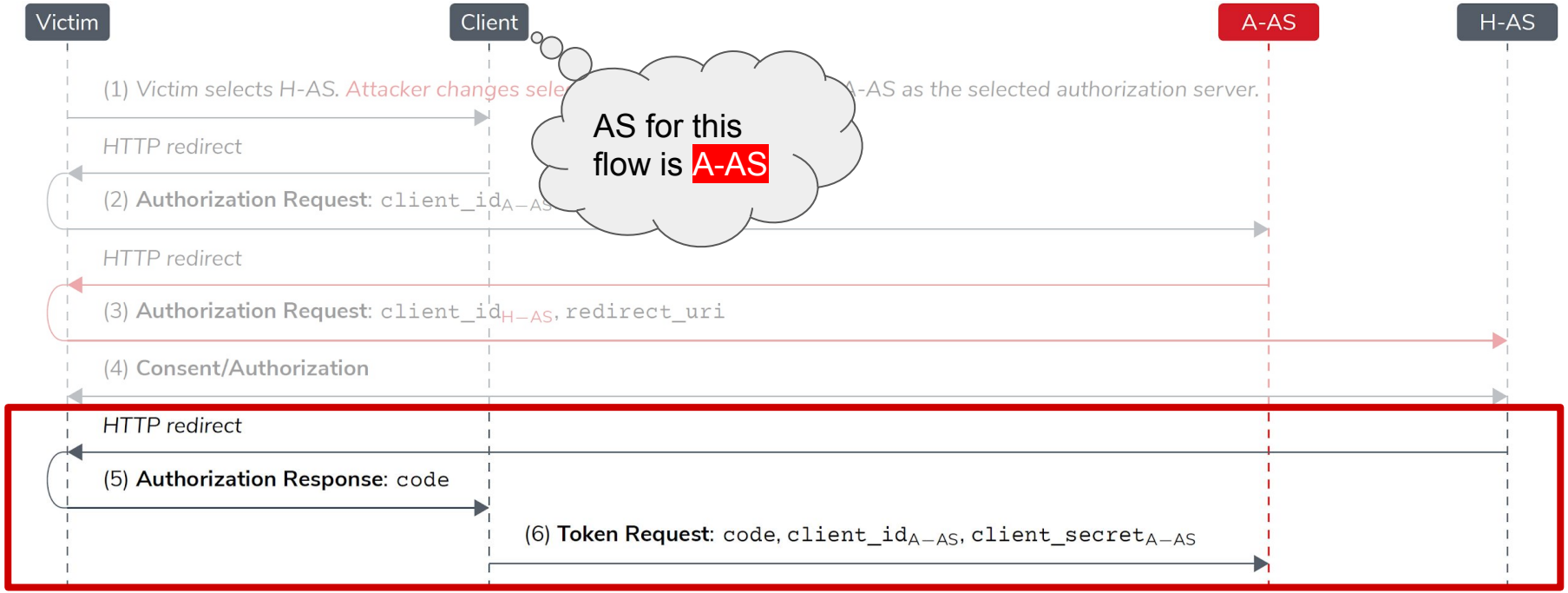
Does PKCE help?

With PKCE: Correct code verifier required to redeem code. **PKCE Chosen Challenge Attack:**

1. Attacker **takes `code_challenge` from second session** with the same client and H-AS,
2. **injects** it into the forged authorization request, and
3. runs a **code injection attack as before** (his client will use correct code_verifier).



The Core of Mix-Up Attacks



Idea: Add “Source Identifier” to Auth Response

Add information about the AS to the authorization response.

Using existing mechanisms:

- Clients register a separate redirect URI for each AS
- AS matches full redirect URI against registered URI (no variable parts)
- Clients match URI of authorization response and AS’s redirect URI

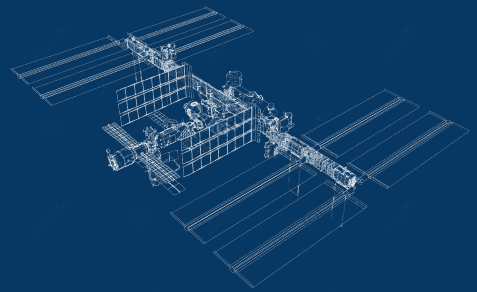
Per-AS Redirect URIs: Problems

While only using existing mechanisms, this solution...

- **… requires a lot of care at the client's side.**
  E.g., how to encode and manage AS identifiers.
- **… has subtle pitfalls.**
  E.g.: redirect URI must be unique for each combination of (authorization endpoint URI, token endpoint URI).
- **… is not suitable for ecosystems with centralized client registration.**
- **… can be circumvented:**
  - With dynamic client registration:
    Attacker-AS can modify registered redirect URI to use same as H-AS.
  - In combination with client impersonation:
    Attacker registers new client at H-AS with the redirect URI of A-AS.

Robust Solution: `iss` Parameter.

`iss`: Technical Overview



The `iss` Parameter

- Idea: Add issuer identifier (as defined in RFC8414) to authorization response
- Example authorization response:

```
HTTP/1.1 302 Found
Location: https://client.example/cb?
  code=x1848ZT64p4IirMPT0R-X3141MFPTuBX-VFL_cvaplMH58
  &state=ZWVlNDBlYzA1NjdkMDNhYjg3ZjUxZjAyNGQzMTM2NzI
  &iss=https%3A%2F%2Fhonest.as.example
```

- Enables the client to determine who issued the authorization response

The `iss` Parameter

- AS supporting this specification MUST add the `iss` parameter to all authorization responses, including error responses
- Example error response:

```
HTTP/1.1 302 Found
Location: https://client.example/cb?
  error=access_denied
  &state=ZWVlNDBlYzA1NjdkMDNhYjg3ZjUxZjAyNGQzMTM2NzI
  &iss=https%3A%2F%2Fhonest.as.example
```

Providing the Issuer Identifier

- AS MUST provide its issuer identifier
- If AS metadata is used:
  - `iss` parameter MUST be identical to AS metadata
  - AS MAY provide issuer identifier additionally by other means (out of scope)
- If AS metadata is not used:
  - Use deployment-specific ways to provide identifier (e. g. static configuration)

Validation of the Issuer Identifier

- Clients MUST compare `iss` parameter to issuer identifier of the AS where the authorization request was sent to
  - MUST reject authorization response if they do not match
- If AS metadata is not used:
  - e. g. use statically configured expected `iss` value for each AS
- Clients MUST NOT allow multiple AS to use the same issuer identifier during registration or configuration

Authorization Server Metadata

- `authorization_response_iss_parameter_supported`
  - Boolean value indicating whether the authorization server provides the `iss` parameter in the authorization response.

Security Considerations

Is this Secure?

Most likely, yes:

Security of the iss parameter against mix-up attacks was proven in a formal web model.

Usual disclaimer: Models make certain assumptions.

Daniel Fett, Ralf Kuesters, Guido Schmitz: A Comprehensive Formal Security Analysis of OAuth 2.0, https://arxiv.org/abs/1601.01229

Should the `iss` parameter be integrity protected?

- JARM could be used to protect authorization response
- Reminder: Client receives authorization response from honest AS

- If the attacker can tamper the authorization response he has direct access to the code and does not need a mix-up attack

Answer: Integrity protection is not necessary for mix-up prevention.

Correlation with JARM and OIDC

- Alternative countermeasures to mix-up attacks are possible
- If issuer identifier is already included in authorization response, `iss` MAY be omitted
  - Examples:
    - OpenID Connect hybrid flow (`response_type=code id_token`)
      - `iss` in ID token
    - JWT Secured Authorization Response Mode (JARM)
      - `iss` in JWT response document
  - If an authorization response contains multiple issuer identifier the client must reject the response if these identifiers do not match
  - If JARM is used, `iss` parameter MUST NOT be used (JARM forbids additional parameters)

Mix-Up Mitigation and the Security BCP

So far, draft-ietf-oauth-security-topics recommends/mandates

1. precise redirect URI checking + per-issuer redirect URIs
2. or non-standard iss parameter.

Target: Make (2.) the default and provide a standard for it.

Details TBA.

Implementations

Implementations of the `iss` Parameter

- yes® ecosystem
- Support in connect2id since version 10.2
- Positive feedback from other implementers

Next Steps

Next Steps

- Working Group Adoption
- Further Feedback