

# GNAP Meeting

## IETF 113

draft-ietf-gnap-core-protocol-09  
draft-ietf-gnap-resource-servers-01

March 25, 2022

Justin Richer • Aaron Parecki • Fabien Imbault

# Agenda

- Core draft update: changes since IETF112 (from -08 to -09)
  - Editorial Changes
  - Functional Changes
- RS draft update: no changes since IETF111 (-01)
  - A handful of small changes accepted but not published
- Draft roadmap: process issue backlog

Differences since IETF112 (Core: -08 to -09)

<https://www.ietf.org/rfcdiff?url2=draft-ietf-gnap-core-protocol-09&url1=draft-ietf-gnap-core-protocol-08>

<https://www.ietf.org/archive/id/draft-ietf-gnap-resource-servers-01.html>

# 37 (core) & 2 (RS) Merged Pull Requests

<https://github.com/ietf-wg-gnap/gnap-core-protocol/pulls>

[?q=is%3Aclosed+closed%3A2021-10-26..2022-03-07](https://github.com/ietf-wg-gnap/gnap-core-protocol/pulls?q=is%3Aclosed+closed%3A2021-10-26..2022-03-07)

<https://github.com/ietf-wg-gnap/gnap-resource-servers/pulls>

[?q=is%3Aclosed+closed%3A2021-10-26..2022-03-07](https://github.com/ietf-wg-gnap/gnap-resource-servers/pulls?q=is%3Aclosed+closed%3A2021-10-26..2022-03-07)

# 40 (core) closed issues

<https://github.com/ietf-wg-gnap/gnap-core-protocol/issues>

[?q=is%3Aissue+is%3Aclosed+closed%3A2021-10-26..2022-03-07](https://github.com/ietf-wg-gnap/gnap-core-protocol/issues?q=is%3Aissue+is%3Aclosed+closed%3A2021-10-26..2022-03-07)

No closed issues on the RS draft

# Editorial Changes

- Text consistency:
  - [#387](#), [#386](#)
- Editorial:
  - [#413](#), [#403](#), [#400](#), [#394](#), [#384](#), [#377](#), [#358](#), [#357](#), [#340](#), [#339](#)
- Release and cleanup:
  - [#414](#), [#381](#), [#365](#), [#359](#), [#356](#), [#355](#), [#297](#)

# Functional Changes

- Security Considerations: [#402](#), [#367](#), [#360](#), [#351](#)
- Subject Identifier: [#401](#), [#379](#)
- Keys: [#399](#), [#362](#)
- Discovery: [#398](#)
- Interaction: [#390](#), [#389](#), [#388](#), [#380](#)
- Error Codes: [#385](#)
- Token Management: [#383](#), [#366](#)

# Editorial Changes

- Use of “URI” instead of “URL”
- Use of “end user” instead of “end-user”
- Consistent formatting of requirements in parameter lists



# User Code Interactions

- Previously user\_code MAY include a URI that SHOULD NOT vary (but it might vary anyway?)
- Now split into “user\_code” and “user\_code\_uri”

Old mode:

```
{
  "user_code": {
    "code": "ABC1GHF",
    "url": "https://srv.go/dev"
  }
}
```

New modes:

```
{
  "user_code": {
    "code": "ABC1GHF"
  },
  "user_code_uri": {
    "code": "09KLBA231",
    "uri": "https://srv.go/dev"
  }
}
```

# Why change this?

- Previous mode was ambiguous
  - The URL shouldn't change, but then why send it?
  - If it does change, can the client safely ignore it?
- Now the client can select modes based on its abilities
  - Can't display a URL? Use "user\_code"
  - Can display a URL? Use "user\_code\_uri"

# Open Questions

- Does the “user\_code” need to be an object anymore or can we collapse this into a single value?
- Does the “redirect” mode name still make sense?

Proposal:

```
{  
  "user_code": "ABC1GHF",  
  "arbitrary_uri": "https://server.example.com/093rj43t0"  
}
```

# Subject information request

- Renamed fields

Old mode:

```
"subject": {  
  "formats":  
    ["iss_sub", "opaque"],  
  "assertion_formats":  
    ["id_token"]  
}
```

New modes:

```
"subject": {  
  "sub_id_formats":  
    ["iss_sub", "opaque"],  
  "assertion_formats":  
    ["id_token"]  
}
```

# Subject information response

- Changed structures (mostly for assertions)
- Parallel

Old mode:

```
"subject": {
  "sub_ids": [ {
    "format": "opaque",
    "id": "J2G8G804AZ"
  } ],
  "assertion": {
    "id_token": "ejy..."
  }
}
```

New modes:

```
"subject": {
  "sub_ids": [ {
    "format": "opaque",
    "id": "J2G8G804AZ"
  } ],
  "assertions": [ {
    "format": "id_token",
    "value": "ejy..."
  } ]
}
```

# Security considerations

- Redirect codes
- Session management
- Stolen token replay
- Self-contained access tokens
- Network problems during token management
- Server-side request forgery

# Preventing Mix-Up Attacks

- Clients SHOULD use a different key for each AS they talk to

A client instance that is capable of talking to multiple AS's SHOULD use a different key for each AS to prevent a class of mix-up attacks as described in Section 12.28.

# User presence

- Used to have a normative requirement for user to be there during callback
  - Not testable or enforceable in a meaningful way
  - Confusing for developers (what am I supposed to do here?)
- Removed requirements and added security considerations
  - Expanded discussions on session management
  - Encourage strong session linking and explain why
  - Call out tradeoffs for polling (no “finish” method)



# Error Responses

- Added new section for programmatic error responses from AS
- Error codes:
  - "invalid\_request": The request is missing a required parameter, includes an invalid parameter value or is otherwise malformed.
  - "invalid\_client": The request was made from a client that was not recognized or allowed by the AS, or the client's signature validation failed.
  - "user\_denied": The RO denied the request.
  - "too\_fast": The client instance did not respect the timeout in the wait response.
  - "unknown\_request": The request referenced an unknown ongoing access request.
  - "request\_denied": The request was denied for an unspecified reason.

# Token Rotation

- Rotating an access token through management URI replaces that token
  - Old token is effectively thrown out if possible
- Making a request to the grant continuation URI creates a new token
  - Old token might be thrown out, or not
- Will clarify more with proposed “grant lifecycle” discussion