GNAP Meeting
IETF 112

draft-ietf-gnap-core-protocol-08
draft-ietf-gnap-resource-servers-01

November 11, 2021

Justin Richer • Aaron Parecki • Fabien Imbault

Agenda

- Core draft update: changes since IETF111 (from -06 to -08)
  - Editorial Changes
  - Functional Changes
- RS draft update: no changes since IETF111 (-01)
  - A handful of small changes accepted but not published
- Formal security analysis
- Draft roadmap: process issue backlog

Differences since IETF111 (Core: -06 to -08)

https://www.ietf.org/rfcdiff
?url2=draft-ietf-gnap-core-protocol-08
&url1=draft-ietf-gnap-core-protocol-06

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

32 (core) & 3 (RS) Merged Pull Requests

https://github.com/ietf-wg-gnap/gnap-core-protocol/pulls?q=is%3Aclosed+closed%3A2021-07-13..2021-10-25

https://github.com/ietf-wg-gnap/gnap-resource-servers/pulls?q=is%3Aclosed+closed%3A2021-07-13..2021-10-25

55 (core) & 5 (RS) closed issues

https://github.com/ietf-wg-gnap/gnap-core-protocol/issues?q=is%3Aissue+is%3Aclosed+closed%3A2021-07-13..2021-10-25

https://github.com/ietf-wg-gnap/gnap-resource-servers/issues?q=is%3Aissue+is%3Aclosed+closed%3A2021-07-13..2021-10-25

Editorial Changes

- Text consistency: 308, 313, 314, 315, 316, 318, 319, 321, 323, 324, 325, 328, 331
- Editorial: 310, 311, 312, 327, 335, 336
- Editorconfig: 294
- Contributors: 320
- Release and cleanup: 338

Functional Changes

- Trust relationships: 306, 337
- Security considerations: 304, 317, 330
- Privacy considerations: 307, 332
- Subject identifier: 305, 308
- Client instance identifier: 333

Trust Relationships

- Defined using promise theory (new informative reference)
  - Allowing for a formal trust model, including threats
- New section 1.4 details the promises between end-user/RO, end-user/client, end-user/AS, client/AS, RS/RO, AS/RO, AS/RS
- Refers to security and privacy considerations

$$A_1 \overset{b}{\textbf{Trusts}} A_2. \tag{10.4}$$

In this case, trust is seen to be a dual concept to that of a promise. If we use the notation of ref. [BFb], then we can write trust as one possible valuation $v : \pi \to [0, 1]$ by A_1 of the promise made by A_2 to it:

$$A_1[A_2] \overset{b}{\textbf{Trusts}} A_2[A_1] \leftrightarrow v_1(A_2 \overset{b}{\to} A_1) \tag{10.5}$$

This is then a valuation on a par with economic valuations of how much a promise is worth to an agent[BFb]. The recipient of a promise can only make such a valuation if it knows that the promise has been made.

Proposal 1 (Trust). *An* agent's expectation *that a promise will be kept. It may be assigned a value lying between 0 and 1, in the manner of a Bayesian probability.*

Proposal 2. *Trust of an agent S by another agent R can exist if agent R is informed that agent S has made a promise to it in the past, or if the recipient of the promise R is able to infer by indirect means that S has made such a promise.*

Security Considerations

- 25 Subsections, including:
  - TLS is required, and you also have to sign things
  - You have to protect your keys and other artifacts
  - Bearer tokens cause problems
  - Use real crypto and randomization
  - Front-channel redirects are inherently susceptible to attack
  - You have to check all the hashes and signatures
  - Pre-registration doesn’t solve all the problems you think it does
  - MTLS doesn’t solve all the problems you think it does
  - TLS can be deployed in a few different ways
  - Just because something is signed doesn’t mean you can trust it
  - Processing assertions can be complex if you do it wrong (esp. SAML)

Privacy Considerations

- Modeled after RFC6973
- Main topics:
  - Surveillance
    - Surveillance by the Client
    - Surveillance by the Authorization Server
  - Stored Data
  - Intrusion
  - Correlation
    - Correlation by Clients
    - Correlation by Resource Servers
    - Correlation by Authorization Servers
  - Disclosure in Shared References

Symmetric Cryptography

- Allowed but restricted:
  - Underlying crypto methods allow for symmetric cryptography
  - GNAP does not allow for symmetric key **distribution**
    - Only identifiers can get passed around
  - KMS and key derivation are safe practices
  - Post-quantum cryptography is largely symmetric
- Security considerations and normative requirements limit its use

User Handle

- Use “subject information” opaque identifier instead of separate user handle
- Simplifies the protocol, uses constructs we already have

Response from AS:

```
{
  "subject": [{
    "format": "opaque",
    "id": "XUT2MFM1XBIKJKSDU8QM"
  }]
}
```

Request from Client Instance:

```
{
  "user": "XUT2MFM1XBIKJKSDU8QM"
}
```

(or)

```
{
  "user": [{
    "format": "opaque",
    "id": "XUT2MFM1XBIKJKSDU8QM"
  }]
}
```

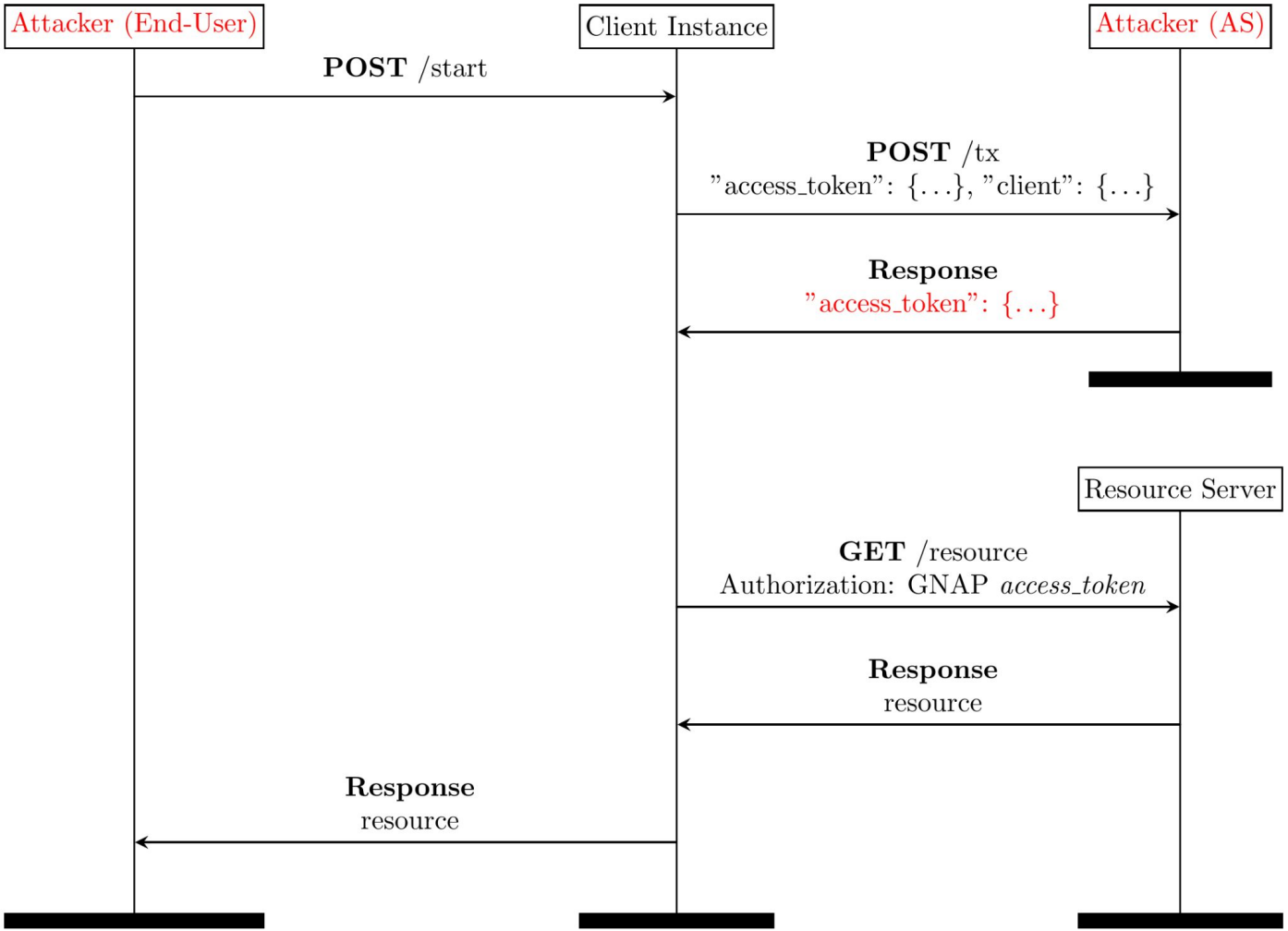
Removed “handle” discussion

- AS used to return many different “handles” for different purposes
  - “user_handle” -> now “opaque” identifier
  - “resource_handle” -> now from RS
- Now only client “instance_id”
  - Could this be simplified further?

Formal GNAP Security Analysis

Cuckoo Token Attack

- Client instance talks to two AS’s
  - Uses the same keys on both
  - Tricked into using attacker’s AS to get token for RS
- Attacker steals key-bound token and replays it from their own AS
- Attacker gets client instance to use bound token at honest RS


Attacker (End-User)
Client Instance
Attacker (AS)
POST /start
POST /tx
"access_token": {...}, "client": {...}
Response
"access_token": {...}
Resource Server
GET /resource
Authorization: GNAP access_token
Response
resource
Response
resource

Proposed Mitigations

- Client instance sends AS identifier alongside access token
  - RS now has to check these are consistent
  - Client has to send more data each time
  - (Protocol change)
- Client instance uses different keys with each AS
  - Stolen token bound to different keys, RS will reject
  - (Security consideration)
- Client has strong binding between RS and AS used
  - Attacker can’t convince client to use “wrong” AS
  - (Security consideration)

307 Redirect Attack

- HTTP 307 causes POST to be re-POSTed
- Can leak important information from front-channel session to back-end components
- Recommended mitigation:
  - Security consideration discussion
  - Normative requirements on redirection-based interaction functions

Discussion Items

Draft Roadmap

- **Process the issue backlog**
  - https://github.com/ietf-wg-gnap/gnap-core-protocol/issues
  - https://github.com/ietf-wg-gnap/gnap-resource-servers/issues
- Clarity on what's allowed/not allowed at each step
- Key rotation
- Mandatory to Implement
- Extension discussion
  - IANA Registries
- What to do with JOSE
- Focus on the RS Draft

Clarity on what’s allowed at each step

- Open questions:
  - Can you send “client” on a continuation request?
  - Can you send “interact_ref” multiple times?
  - Do you need to only use a “redirect” start method once, or can you do it multiple times?
- Editors have probable answers, will propose text to close these

Key Rotation Proposal

- WG feedback: feature is desirable
- Use different mechanisms for each presentation type
  - HTTPSig: multiple signatures
  - MTLS: PKI cert management
  - JOSE: wrapped JOSE objects
- Apply equally to each place that needs it
  - Client instance keys
  - Access token keys
- Reuse existing infrastructure and tooling where possible

Mandatory to Implement

- GNAP is very flexible (by design)
  - But most of the optional functions are negotiated at runtime
  - Always start the same way, can always get an answer (even if it's "no")
- What is the set of features/functions that are MTI
  - For an AS?
  - For a client instance?
  - For an RS?
- Should we have interoperability profiles?
  - "Redirect-based web app"
  - "Mobile app with launch URL"
  - "Embedded device with polling"

Extensions

- What can be extended?
  - New fields in request and response
  - New data types for existing fields?
- Are extensions ignored if unknown?
- Ensure extensions don’t break the core
- Other general-purpose extension mechanisms:
  - End-user claim requests (VCs? OIDC?)
  - ‘access’ types (already discussed)
- Interaction start/finish mechanisms
  - And how they combine

JOSE

- Two JOSE-based key-proofing mechanisms kept in core
  - Detached JWS header
  - Attached JWS (replaces request body, when possible)
- Only JOSE dependencies in GNAP core
- Should these be their own spec?
- Could they be used outside of GNAP?

RS Draft: Future work

- Security/Privacy/Trust considerations
- Token model
  - Not a token format!

Implementation

Implementation status

- Java implementation updated to latest draft
  - Python, PHP, and Rust in the works
- Dependency implementations:
  - HTTP Message Signatures implementations (Java, Python, Go)
  - SECEVENT identifier implementations (Java, Python, JS, Rust)
- Editors will add an implementation status section to core draft
- Major churn is still quiet
  - Some syntax and details are still being bikeshedded
  - Dependency churn has also died down

Open Discussion