Avoiding Gaps in Authorization Solutions for the Internet of Things

Stefanie Gerdes, Olaf Bergmann, **Carsten Bormann**

NDSS DISS Workshop 2018

Authorization in the IoT

- IoT: varied hardware with varied constraints:
  - limited processing power, storage space, RAM...
  - lack of user interfaces and displays
  - limited ability to relate to wall-clock or even measure time
- M2M: smart objects often need to communicate without user interaction
- perimeter security not sufficient (e.g., wireless communication)
- smart objects must be enabled to enforce their owners' security policies

Authenticated Authorization

- overseeing principal: the person/organization that defines security policies for a certain endpoint (owner, user, . . .)
- overseeing principals are the main authorities for their data and devices: their decision about if and how these assets are used must be followed
- smart objects require certain information for authorization
- attackers may try to manipulate claims
- claims that influence the authorization must stem from claimants authorized by the overseeing principal

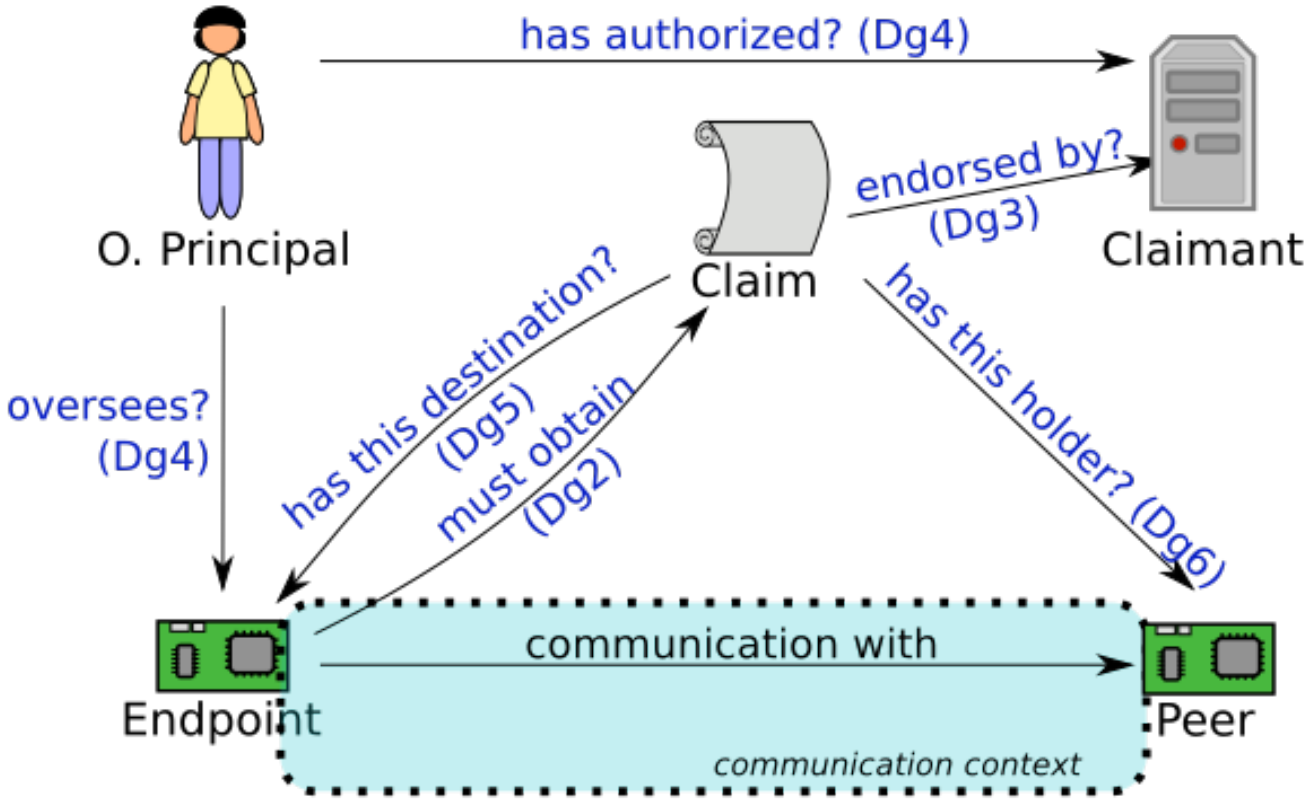
Gaps in Authorization Solutions

- finding vulnerabilities in security solutions may take a while
  - e.g., missing timestamps in Needham-Schroeder protocol
- lack of explicitness (of, e.g., requirements) leads to gaps
  - implementers must fill in their own interpretation
- effective authorization requires continuous protection
- how to avoid gaps in protocols?
  - make necessary characteristics explicit
  - define checklist of tasks that endpoints must perform
  - perform tasks for every sent and received piece of information

Claims

- Must comprise or relate to all relevant information of the communication context:
  - claim statement (what does the claim state?),
  - claim destination (who is supposed to get the claim?),
  - claim holder (who is the holder of the claim?).
- All pieces of information must be bound together and endorsed by the claimant.

Delegation Tasks Overview



Delegation Tasks

- Dg0 input and output of a claim only from/to authorized entities
- Dg1 statement (e.g., peer attributes), destination and holder must be set and endorsed
- Dg2 all currently valid claims must be securely obtained
- Dg3 the endorsement of the claim must be validated (e.g., check signature)
- Dg4 the claimant's authorization must be validated
- Dg5 validate the intended destination of the claim (is it for me?)
- Dg6 validate holder (is my current peer the holder?)
- Dg7 evaluate claim (how to handle multiple currently valid claims?)

Task Omission

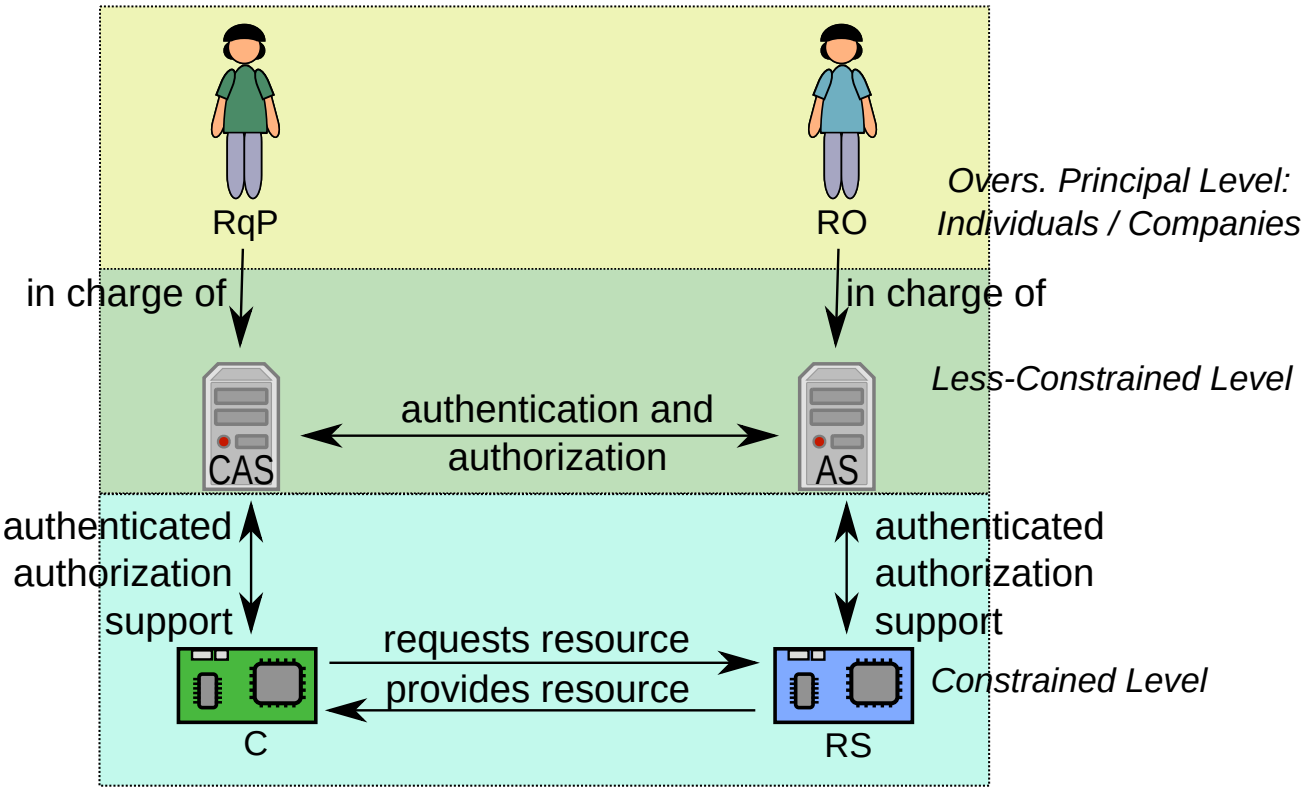
What happens if Tasks are omitted?

- Dg0 Delegator Authz: disclosure of confidential information.
- Dg1 Claim Config: claim is wrongly configured.
- Dg2 Obtainment: critical information is missing.
- Dg3 Binding & Endorsement: attackers can manipulate or forge claims.
- Dg4 Authorization: unauthorized entities can issue claims.
- Dg5 Destination: man-in-the-middle attacks.
- Dg6 Holder: unclear about whom the claim is.
- Dg7 Evaluation: critical information is not considered.

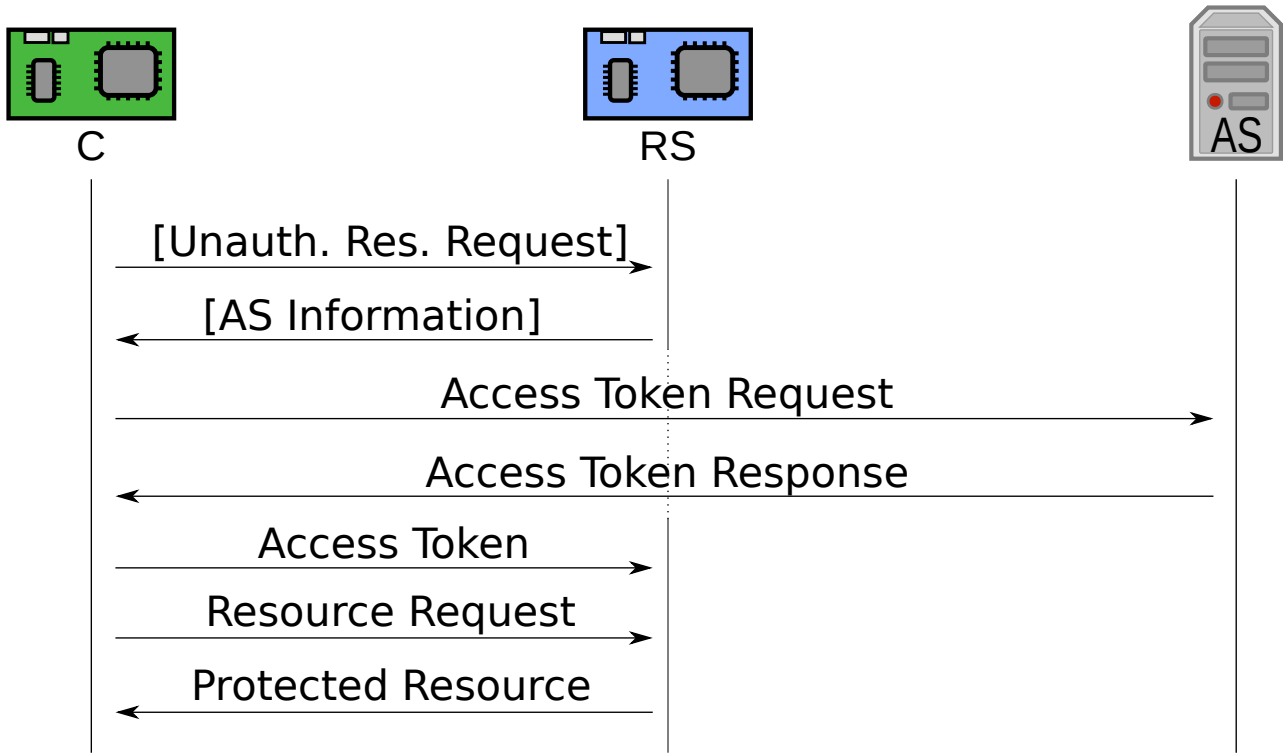
Example: ACE DTLS Profile

- IETF ACE WG: Authentication and Authorization in Constrained Environments
- ACE framework: derived from OAuth 2.0 framework
- Define profiles to add details to obtain actual protocol
- DTLS profile defines communication over DTLS

ACE Architecture



ACE Framework Protocol Flow



Analysis Example: Validity

- Task Dg2: endpoint must obtain and use the most recent claims about peer.
- RS must check if access token is still valid.
- Framework provides three options:
  1. Token contains expiration time (requires RTC + secure time)
     - What if multiple valid access tokens exist (e.g., token updates).
     - Expiration time not useful to distinguish tickets.
  2. RS asks AS about token validity
     - How does RS determine if the response from AS is fresh (e.g., timeouts)?
  3. Incremental sequence numbers for tokens (RS discards tokens with old sequence numbers)
     - C may decide to not relay tokens to RS. Old tokens are then infinitely valid.

Analysis Example: RS Mismatch

- For task Dg1: refer to RS's attributes in RS information
- In ACE framework only through communication context: C specifies RS in AS request, AS response must belong to request (not explicitly mentioned — how is this binding made secure?)
- How does C specify RS?
  - IP address? May change → C communicates with wrong RS
  - Identifier specified by AS? → how does C get to know RS's identifier?
- Without common understanding about RS, C may communicate with the wrong RS.
- **C cannot detect if it is communicating with wrong RS.**

Server (RS) Support Summary

- scope is not mandatory. purpose of access token without scope is unclear, can be misinterpreted (Dg1)
- integrity-protection of access tokens unclear (Dg1)
- freshness and validity of tokens require more work to support constrained RSs (Dg2)
- unclear how RS handles unprotected access tokens (Dg3)
- unclear how RS handles access tokens where essential fields are missing (Dg3)
- unclear if RS checks that the access token stems from an AS with which RS has registered (Dg4)
- unclear how RS handles multiple valid access tokens (Dg7)
- all fields in access tokens are optional $\rightarrow$ essential information may be missing.
- Mitigation: implementers will get this right (yeah, right)

Client Support Summary

- C may trick AS to issue access tokens with C's permissions and other client's RPKs $\rightarrow$ give own permissions to other clients without disclosing C's own credentials (Dg0)
- AS may accidentally give keying material for wrong RS to C (Dg1+Dg3).
- a constrained client cannot determine the validity of the RS information (Dg2)
- static list of authorized ASs may become outdated, must be kept up to date by RqP (Dg4)
- the connection between AS and C may be vulnerable to MitM attacks (Dg5)
- How C enforces RqP's authorization rules is not addressed $\rightarrow$ C may communicate with unauthorized RSs and violate RqP's and even RO's security objectives (Dg6).

Open Questions

- Only RS gets authorization information from RO
- RqP's security objectives are currently not considered
- Unsupervised clients cannot actively participate in protection, must rely on RS.
  - What if RS is not authorized by RqP?
  - What if RqP and RO have distinct authorization policies?
- ACE framework requires every client to register with every AS
  - close coupling across company borders (RESTful? Scalability?)

Summary

- Delegation task analysis can detect gaps and vulnerabilities in authorization solutions.
- Gaps in DTLS profile and ACE framework need to be fixed.
- Client support should be improved.
- Unsupervised clients should be supported by implementing the four-corner architecture (C has own less-constrained device).
- Protocol Designers must ascertain the explicitness of their specifications (use the checklist!)


MIND THE GAP