Delegated Authenticated Authorization for Constrained Environments

Stefanie Gerdes, Olaf Bergmann, Carsten Bormann
{gerdes|bergmann|cabo}@tzi.org
Universität Bremen

NPSec’14, 2014-10-21
Motivation

- Smart objects
  - small devices with specific purpose
  - low cost, limited abilities

- Internet of Things
  - interconnect things and their users to enable new applications
  - 50 billion connected devices expected by 2020 (Cisco)

- Smart objects are expected to be integrated in all aspects of everyday life
  - entrusted with vast amounts of data important to our lives.
  - need to communicate unseen and autonomously.
Limitations of “Constrained Environments”

- processing power
- storage space
- network capacities
- lack of user interfaces and displays
- energy (often powered from primary batteries)

- RFC 7228: Terminology for Constrained-Node Networks
  - device classification
  - energy profile
  - sleep strategies
# Classes of Constrained Devices

<table>
<thead>
<tr>
<th>Name</th>
<th>data size (e.g., RAM)</th>
<th>code size (e.g., Flash)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 0, C0</td>
<td>&lt;&lt; 10 KiB</td>
<td>&lt;&lt; 100 KiB</td>
</tr>
<tr>
<td>Class 1, C1</td>
<td>~ 10 KiB</td>
<td>~ 100 KiB</td>
</tr>
<tr>
<td>Class 2, C2</td>
<td>~ 50 KiB</td>
<td>~ 250 KiB</td>
</tr>
</tbody>
</table>

Source: RFC 7228
# Classes of Constrained Devices

<table>
<thead>
<tr>
<th>Name</th>
<th>data size (e.g., RAM)</th>
<th>code size (e.g., Flash)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 0, C0</td>
<td>&lt;&lt; 10 KiB</td>
<td>&lt;&lt; 100 KiB</td>
</tr>
<tr>
<td><strong>Class 1, C1</strong></td>
<td>~ 10 KiB</td>
<td>~ 100 KiB</td>
</tr>
<tr>
<td>Class 2, C2</td>
<td>~ 50 KiB</td>
<td>~ 250 KiB</td>
</tr>
</tbody>
</table>

Source: RFC 7228
Communication in Constrained Environments

- Constrained Application Protocol (CoAP, RFC 7252)
  - designed for special requirements of constrained environments
  - Similar to HTTP (RESTful architecture style)
    - server has items of interest
    - client requests representation of current state

- Datagram Transport Layer Security (DTLS) binding
- How can users keep the control over their data and devices?
  -> Authorization
Problem Statement

- A Client (C) wants to access an item of interest, a web resource (R), on a Resource Server (RS).
- A priori, C and RS do not know each other, have no trust relationship. They might belong to different owners.
- C and / or RS are located on a constrained node.

\[ C \xrightarrow{\text{requests resource}} \text{RS} \xleftarrow{\text{provides resource}} \]
Goals of an Authenticated Authorization Protocol in Constrained Environments

- Secure exchange of authorization information
- Establish DTLS channel between constrained nodes
- Use only symmetric key cryptography on constrained nodes
- Support of class-1 devices
- RESTful architectural style
- Relieve constrained nodes from managing authentication and authorization
Authenticated Authorization

- Determine if the owner of an item of interest allows an entity to access this item as requested.
- Authentication: Verify that an entity has certain attributes (cf. RFC4949).
- Authorization: Grant permission to an entity to access an item of interest.
- Authenticated Authorization: Use the verified attributes to determine if an entity is authorized.
Tasks for Authenticated Authorization

Beforehand: Provide information for Authenticated Authorization

- Make attribute-verifier-binding verifiable: Validate that an entity actually has the attributes it claims to have (e.g. that it belongs to a certain user) and bind the attributes to a verifier (e.g. a key) using the endorsement info.
- Define access policies (entity with attribute x has this set of permissions).

At the time of the request: Check access request against the provided information

- Check the verifier a received access request is bound to.
- Check the verifier-attribute binding.
- Determine the authorization using the attributes.
- Enforce the authorization.
Constrained Level Actors

- C and RS are constrained level actors: able to operate on a constrained node.
- C attempts to access a resource.
- RS hosts one or more resources.
- Tasks:
  - Determine if sender is authorized to access as requested.
  - Enforce the authorization
Principal Level Actors

- C and RS are under control of principals in the physical world.
- CO is in charge of C: specifies security policies, e.g. with whom RS is allowed to communicate.
- RO is in charge of RS: specifies security policies, e.g. authorization policies.
A priori, C and RS do not know each other, might belong to different security domains
Constraints

- C and RS
  - are constrained in terms of power, memory, storage space.
  - may not have user interfaces and displays.
  - can only fulfill a limited number of tasks.
  - may not have network connectivity all the time.
  - are not able to manage complex authorization policies.
  - are not able to manage a large number of keys.

- Add another complexity level: less-constrained devices for more difficult tasks
Less-Constrained Level

- AM and AS act in behalf of their respective owner.

- Tasks:
  - Obtain the security objectives from their owner.
  - Authenticate the other party.
  - Provide simplified authorization rules and means for authentication to their constrained devices.
Unauthorized Access Request

**AM**

**AS**

**C**

**RS**

Request

4.01 Unauthorized you should ask AS
Contact RS’s Less Constrained Device for Authorization
Access Ticket

**Access Request**

**Access Ticket**

**Face:**
- authorization info
- timestamp
  - [lifetime]
  - [session key]

**Verifier:**
- session key
Use Access Ticket to Establish DTLS Channel
PSK Derivation

DTLS channel

\(\text{psk\_identity} = \text{Ticket Face}\)

PSK = Verifier

derive PSK from Ticket Face and \(K_{RS,AS}\)
RS Permits Authorized Requests Over DTLS
Initial Trust Relationships

AM  DTLS/TLS  AS

DTLS (PSK)  we have these  DTLS (PSK)

C  RS
Trust: The Complete Picture

DTLS/TLS

AM

AS

DTLS
(PSK)

we have these

C

we want this

RS

DTLS
(PSK)
Evaluation

Reference implementation adds

- about 440 Bytes Code
- 54 Bytes data for ticket face
- 722 Bytes parser for CBOR payload

to existing CoAP/DTLS server (ARM Cortex M3).
Summary: The DCAF Protocol

- Requires less-constrained nodes to do the hard work (possibly including public-key crypto)
- Utilize DTLS to transmit authorization info and access tickets
- Authenticate origin client by its access ticket:
  - RS and AS share at least one session key
  - AS creates Ticket Face + Verifier, tells AM, C
  - C initiates DTLS handshake with RS
  - Ticket Face is PSK identity, Verifier is PSK
  - RS derives PSK from Ticket Face
- Knowledge of Verifier authenticates C to RS!
- Knowledge of PSK authenticates RS to C!
- Authorization information valid for the entire session
- Verifier ensures Face’s integrity
Conclusion

- **Problem**
  - IoT devices may be too constrained to perform Authenticated Authorization
  - enable multi-domain scenario

- **Our solution**
  - Offload complex tasks to less constrained devices
  - use DTLS with symmetric cryptography for secure communication

- **Future Work**
  - Demonstrate interworking of less constrained devices, e.g. using OAuth
  - Define authorization information format for simplified policies
References

- https://tools.ietf.org/id/draft-gerdes-ace-dcaf-authorize