Discovery of OSCORE Groups with the CoRE Resource Directory

draft-tiloca-core-oscore-discovery-08

Marco Tiloca, RISE
Christian Amsüss
Peter van der Stok

IETF 110, CoRE WG, March 8th, 2021
Recap

› A newly deployed device:
  – May not know the OSCORE groups and their Group Manager (GM)
  – May have to wait GMs to be deployed or OSCORE groups to be created

› Use web links for discovery – Typically through the Resource Directory (RD)
  – Discover an OSCORE group and retrieve information to join it
  – Practically, discover the links to join the OSCORE group at its GM
  – CoAP Observe supports early discovery and changes of group information

› Use resource lookup, to retrieve:
  – The name of the OSCORE group
  – A link to the resource at the GM for joining the group

› Full support for both Link-Format and CoRAL RD
Updates from -08

› Added target attributes related the pairwise mode of Group OSCORE
  – ecdh_alg, ecdh_alg_crv, ecdh_key_kty, ecdh_key_crv
  – To refer to for the derivation of pairwise symmetric keys
  – Same advantages as for the attributes on the signature algorithm

› Usage of the right content-format, for links to join OSCORE groups
  – “application/ace-groupcomm+cbor”
  – See Section 8.2 of draft-ietf-ace-key-groupcomm

› Group discovery intended not only to joining nodes
  – Relevant case: signature verifiers, e.g. intermediary gateways
  – They don’t join the OSCORE group, but retrieve public keys from the Group Manager
Updates from -08

› Revised all examples in Link Format and CoRAL

› Proof-of-concept implementation using Link Format [1]
  – The full set of operations from the draft is covered
    › Register an application group
    › Register Group Manager, security group and associated Authorization Server
    › Discover the security group, with descriptive target attributes
    › Discover the associated Authorization Server
    › Discover the application group, with the multicast IP address
  – Successfully tested with Christian’s RD at coap://rd.coap.amsuess.com

Open points

Mostly on security considerations

1. Denial (common issue)
   – The RD hides the presence of groups

2. Interaction leaking (common issue)
   – An endpoint advertises groups and learn addresses of joining nodes as they come
   – Possibly acting also as MITM between joining nodes and real Group Manager

3. Downgrade attack
   – Data from the RD are a hint here
   – Possible to mitigate, by directly checking also with the Group Manager

… and more, consistent with the latest version -27 of the RD
Next steps

› Update also based on the latest draft-ietf-core-resource-directory-27
  – Revised used of the ‘anchor’ attribute
  – Security considerations, covering also the specific open points

› Integrate implementation in the ACE Group Manager and joining node

› Need for more reviews
Thank you!

Comments/questions?

https://gitlab.com/crimson84/draft-tiloca-core-oscore-discovery
Backup
Application/CoAP/Security Groups

› Application group
  – Defined in {RD} and reused as is
  – Set of CoAP endpoints sharing a pool of resources
  – Registered and looked up just as per Appendix A of {RD}

› CoAP Group
  – Defined in draft-ietf-core-groupcomm-bis
  – Set of CoAP endpoints listening to the same IP multicast address
  – The IP multicast address is the ‘base’ address of the link to the application group

› (OSCORE) Security Group
  – Set of CoAP endpoints sharing common security material (e.g. OSCORE Ctx)
  – A GM registers the group-membership resources for accessing its groups
Application vs. Security Groups

Security Group 1
- Application Group 1
- Application Group 2
- Application Group 3

Security Group 2

CoAP group with one multicast address

Different key sets

Resources for given function

Client of application group
Registration

› The GM registers itself with the RD
  – MUST include all its join resources, with their link attributes
  – rt="core.osc.gm", if="ace.group"

Request: GM -> RD

Req: POST coap://rd.example.com/rd?ep=gml
Content-Format: 40
Payload:
</ace-group/feedca570000>;ct=65000;rt="core.osc.gm";if="ace.group";
  sec-gp="feedca570000";app-gp="group1";
  cs_alg="-8";cs_alg_crv="6";
  cs_kenc="1";ecdh_alg="-27";
  ecdh_alg_crv="4",

<coap://as.example.com/token>;
  rel="authorization-server";
  anchor="coap://[2001:db8::ab]/ace-group/feedca570000"

Response: RD -> GM
Res: 2.01 Created
Location-Path: /rd/4521
Discovery (1/2)

› The device performs a resource lookup at the RD
  - Known information: name of the Application Group, i.e. “group1”
  - Need to know: name of the OSCORE Group; Join resource @ GM; Multicast IP address
  - ‘app-gp’ → Name of the Application Group, acting as tie parameter in the RD

Request: Joining node -> RD

Req: GET coap://rd.example.com/rd-lookup/res
  ?rt=core.osc.gm&app-gp=group1

Response: RD -> Joining node

Res: 2.05 Content
Payload:
<coap://[2001:db8::ab]/ace-group/feedca570000>;
ct=65000;
  rt="core.osc.gm"; if="ace.group"; sec-gp="feedca570000";
  app-gp="group1"; cs_alg="-8"; cs_alg_cr"v=6";
  cs_kenc="1"; ecdh_alg="-27"; ecdh_alg_cr"v=4";
  anchor="coap://[2001:db8::ab]"
Discovery (2/2)

The device performs an **endpoint** lookup at the RD
- Still need to know the **Multicast IP address**
- ‘ep’ // Name of the **Application Group**, value from ‘app-gp’
- ‘base’ // Multicast IP address used in the Application Group

Request: Joining node -> RD

```
Req: GET coap://rd.example.com/rd-lookup/ep
    ?et=core.rd-group&ep=group1
```

Response: RD -> Joining node

Res: 2.05 Content
Payload:
```
</rd/501>;ep="group1";et="core.rd-group";
    base="coap://[ff35:30:2001:db8::23]";rt="core.rd-ep"
```
### Alg/key related parameters

- **New optional parameters for a registered group-membership resource**
  - \((*)**(cs_alg : countersignature algorithm, e.g. “EdDSA”
  - \((*) cs_alg_crv : countersignature curve (if applicable), e.g. “Ed25519”
  - \((*) cs_key_kty : countersignature key type, e.g. “OKP”
  - \((*) cs_key_crv : countersignature curve (if applicable), e.g. “Ed25519”
  - \((*) cs_kenc : encoding of public keys, e.g. “COSE_Key”
  - \((*)**(ecdh_alg : ECDH algorithm to derive pairwise keys, e.g. “ECDH-SS + HKDF-256”
  - \((*) ecdh_alg_crv : ECDH curve, e.g. “X25519”
  - \((*) ecdh_key_kty : ECDH key type, e.g. “OKP”
  - \((*) ecdh_key_crv : ECDH curve, e.g. “X25519”
  - \((**) alg : AEAD algorithm, e.g. “AES-CCM-16-64-128”
  - \((**) hkdf : HKDF algorithm, e.g. “HKDF SHA-256”

- **Benefits for a joining node, when discovering the OSCORE group**
  - \((*) No need to ask the GM or to have a trial-and-error when joining the group
  - \((**) Decide whether to join the group or not, based on the supported algorithms