

WGLC / Review changes to constrained join proxy

`draft-ietf-anima-constrained-join-proxy-12`

Michael Richardson, Peter van der Stok, Panos
Kampanakis

IETF 114 - ANIMA Working Group

Discovery issues: GRASP and mDNS Registrar Discovery (by Join Proxy)

Discovery in Constrained-Voucher

```
[M_FLOOD, 51804321, h'fda379a6f6ee00000200000064000001', 180000,  
  [{"AN_join_registrar", 4, 255, "BRSKI_JP"},  
    [0_IPv6_LOCATOR,  
     h'fda379a6f6ee00000200000064000001', IPPROTO_UDP, 5684]]]
```

Arbitrary port

Discovery in Constrained-Join-Proxy

```
[M_FLOOD, 51840231, h'fda379a6f6ee00000200000064000001', 180000,  
  [  
    ["AN_join_registrar", 4, 255, ""], [0_IPv6_LOCATOR,  
     h'fda379a6f6ee00000200000064000001', IPPROTO_TCP, 8443],  
    ["AN_join_registrar", 4, 255, "BRSKI_JP"], [0_IPv6_LOCATOR,  
     h'fda379a6f6ee00000200000064000001', IPPROTO_UDP, 5684],  
    ["AN_join_registrar", 4, 255, "BRSKI_RJP"], [0_IPv6_LOCATOR,  
     h'fda379a6f6ee00000200000064000001', IPPROTO_UDP, 5685]  
  ]]
```

Discovery issues: GRASP and mDNS

Join-Proxy Discovery (by Pledge)

Discovery in Constrained-Voucher

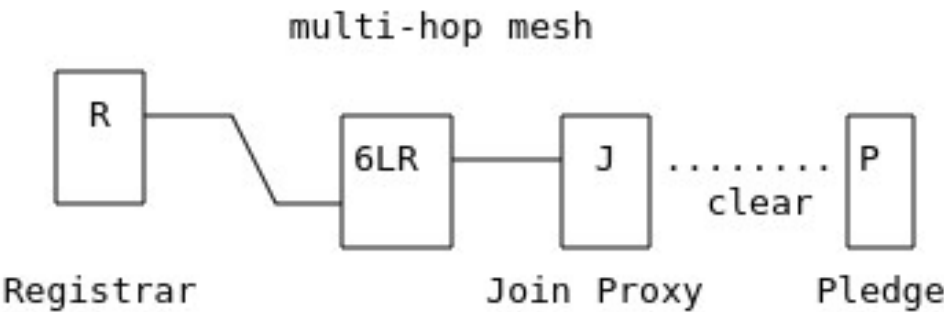
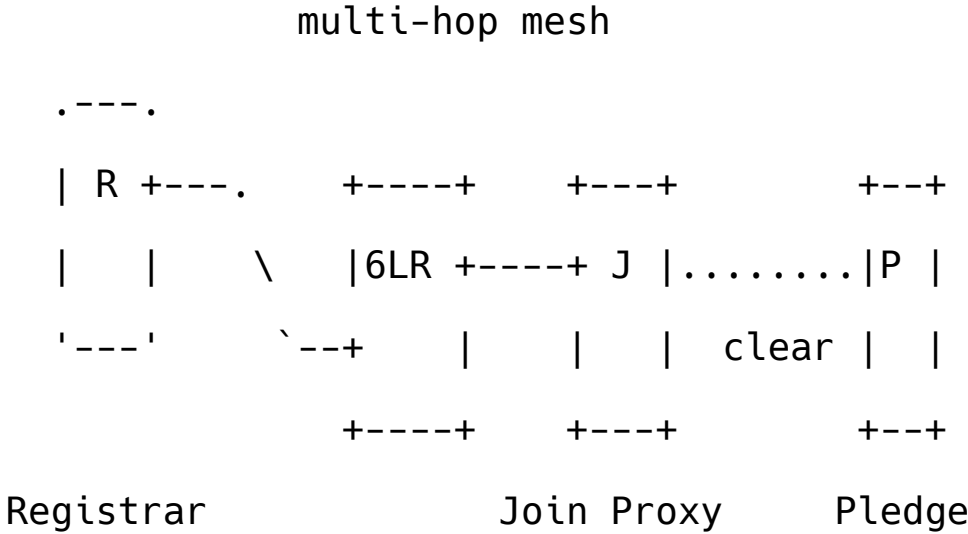
[illegible]

Discovery in Constrained-Join-Proxy

NO CHANGE

Arbitrary port

Mesh Network Diagram



What's Mandatory To Implement?

- Was:
 - “A Join Proxy MAY implement both”
 - Now:
 - “A Join Proxy MUST implement both”
 - Seems to be the result of some review comments.
 - Probably not what we want.
- 1) All Registrars have to support stateful connections, because that's what coaps:// is. They will announce this.
 - 2) Some Registrars support stateless connections (JPY), and those Registrars will announce that.
 - 3) A Join Proxy can support one or both methods. If it supports only stateless, and there is no stateless, then it can not operate as a join proxy. It's not a failure of interoperation, it's a purchasing decision.
 - 4) The goal here is there is no configuration required, not that there is magic that forces every device to implement everything.

| Registrar supports: | Stateful MUST | Stateless (MAY) Registrar does not do Stateless | Stateless (MAY) Registrar does Stateless |
|---------------------------------|--------------------------|---|---|
| Join Proxy Supports: | | | |
| Stateful: YES Stateless: YES | Uses Stateful | Uses Stateful | Uses Stateless |
| Stateful: YES Stateless: NO | Use stateful | N/A | N/A |
| Stateful: NO Stateless: YES | Does not use stateful | • Does not operate as a join proxy | • Uses Stateless |
| Stateful: NO Stateless: NO | • Not a Join Proxy | | |

JPY message changed

Contents SHOULD
be encrypted, but
Contents not standardized

OLD:

```
JPY_message =  
[  
    ip      : bstr,  
    port    : int,  
    family  : int,  
    index   : int  
    content : bstr  
]
```

NEW:

```
JPY_message =  
[  
    pledge_context_message:bstr,  
    content      : bstr  
]
```

Use of CoAP Discovery for JPY “tunnel”

Normal CoAP discovery looks like:

REQ: GET /.well-known/core?rt=brski* <- to Multicast address.

Unicast responses:

RES: 2.05 Content

Content-Format: 40

Payload:

;rt=brski,

</b/rv>;rt=brski.rv;ct=836,

</b/vs>;rt=brski.vs;ct="50 60",

</b/es>;rt=brski.es;ct="50 60"

JPY Discovery looks like this:

REQ: GET /.well-known/core?rt=brski*

RES: 2.05 Content

<coaps://[2001:db8:0:abcd::52]:7634>; rt=brski.rjp,

<coaps://[2001:db8:0:abcd::52]:5683/.well-known/brski/rv>;rt=brski.rv;ct=836,

<coaps://[2001:db8:0:abcd::52]:5683/.well-known/brski/vs>;rt=brski.vs;ct="50 60",

<coaps://[2001:db8:0:abcd::52]:5683/.well-known/brski/es>;rt=brski.es;ct="50 60",

Actually:

- 1) CoAP
- 2) DTLS
- 3) JPY
- 4) UDP
- 5) IPv6

Options for dealing with coaps which is not exactly coaps

- 1) What issue? I don't see an issue, do you?
- 2) Create/Register a new scheme "jpy://"
- 3) Abuse some other scheme (but which one?)
- 4) Never use CoAP Discovery for JPY (GRASP is just fine)
- 5) Your Brilliant Idea Here

Discussion And questions



Current status was AD writeup/reviews

New status: 2nd WGLC?