Approach to the solution space

Implementation status/direction, Cisco AN, IOS/Linux

T. Eckert, M. Behringer, S. Bjarnason, M.Pritkin, BL. Balaji

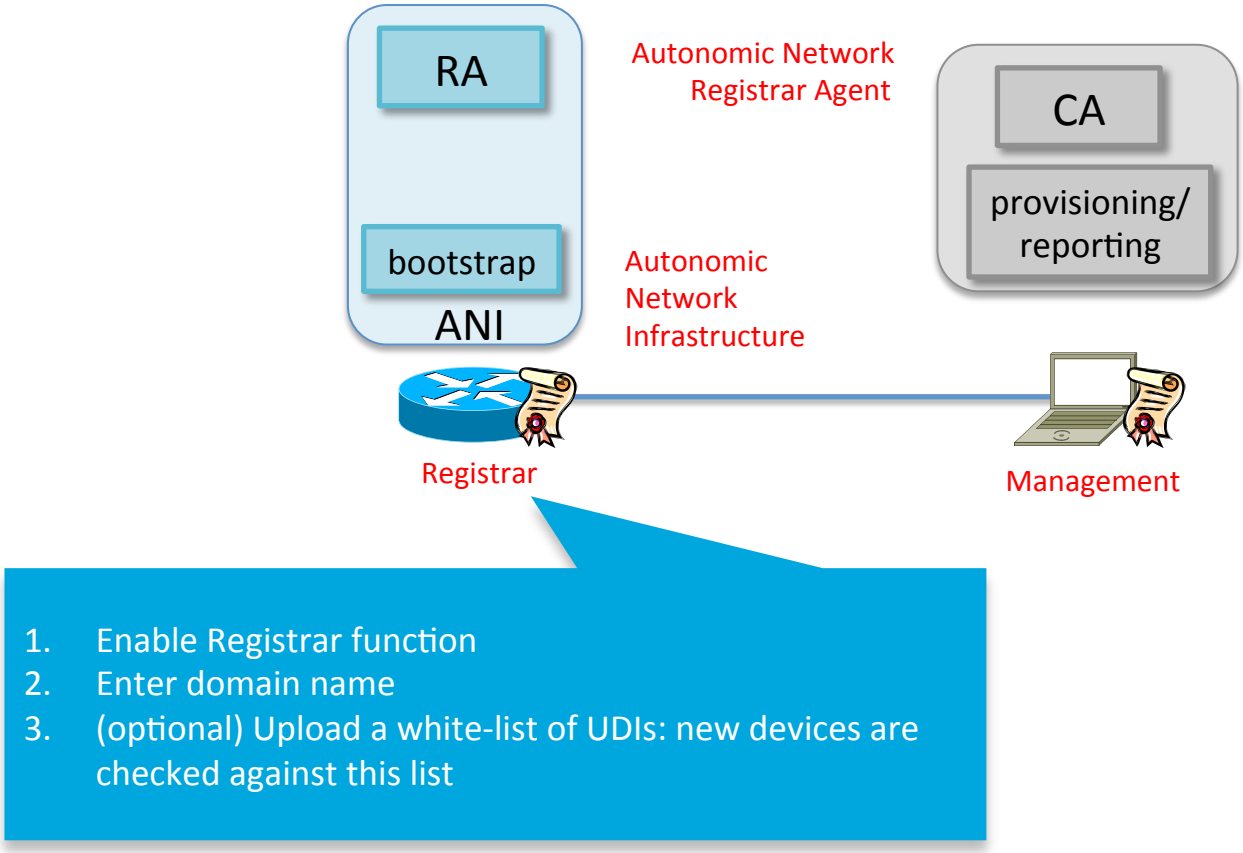
draft-pritikin-bootstrapping-keyinfrastructures [bski]

Solution based on use-case draft-behringer-autonomic-bootstrap-00

draft-behringer-autonomic-control-plane [acp]

Draft is current use-case and solutions draft.

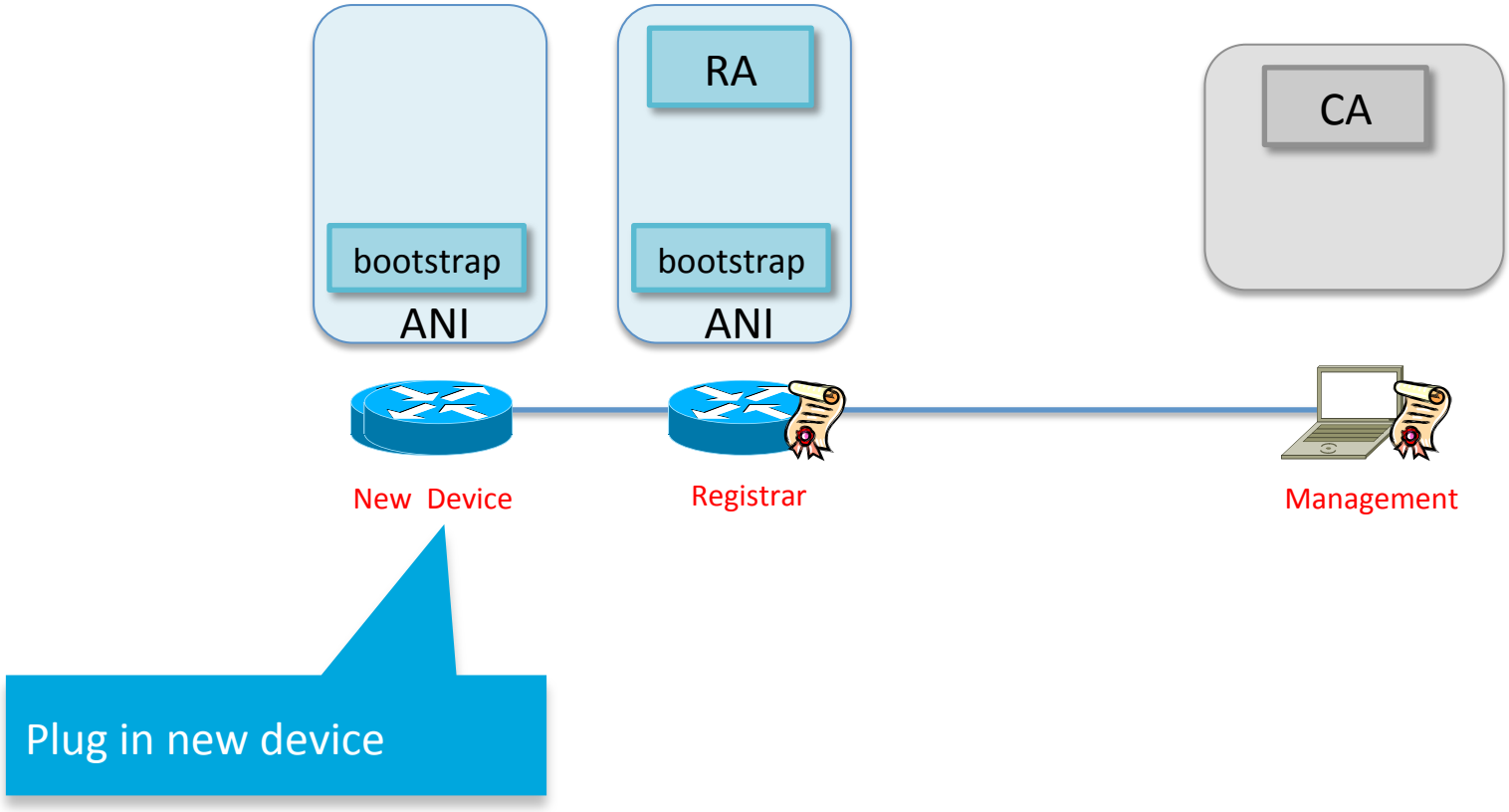
1) Set-up Registrar Device



Notes:

- One registrar is required, multiple support – load-splitting/redundancy supported (for redundancy)
- Registrar only required while new devices join the domain
- Can link into existing CA infrastructure, or be standalone

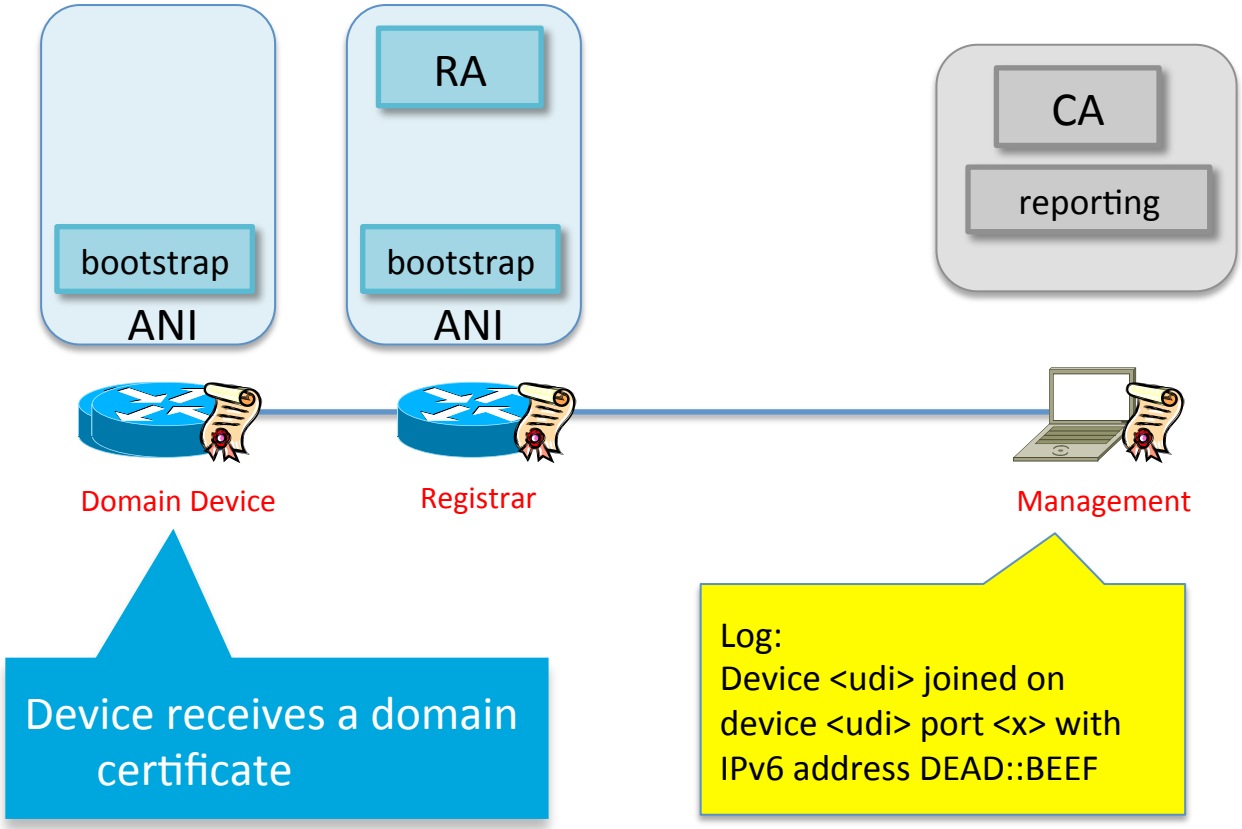
2) Connect a New Device



Notes:

- Complete zero-touch: Device identified by its UDI (later: SUDI)
- If device in white list, no operator intervention at all
- If not, need to manually accept new device based on (S)UDI on registrar

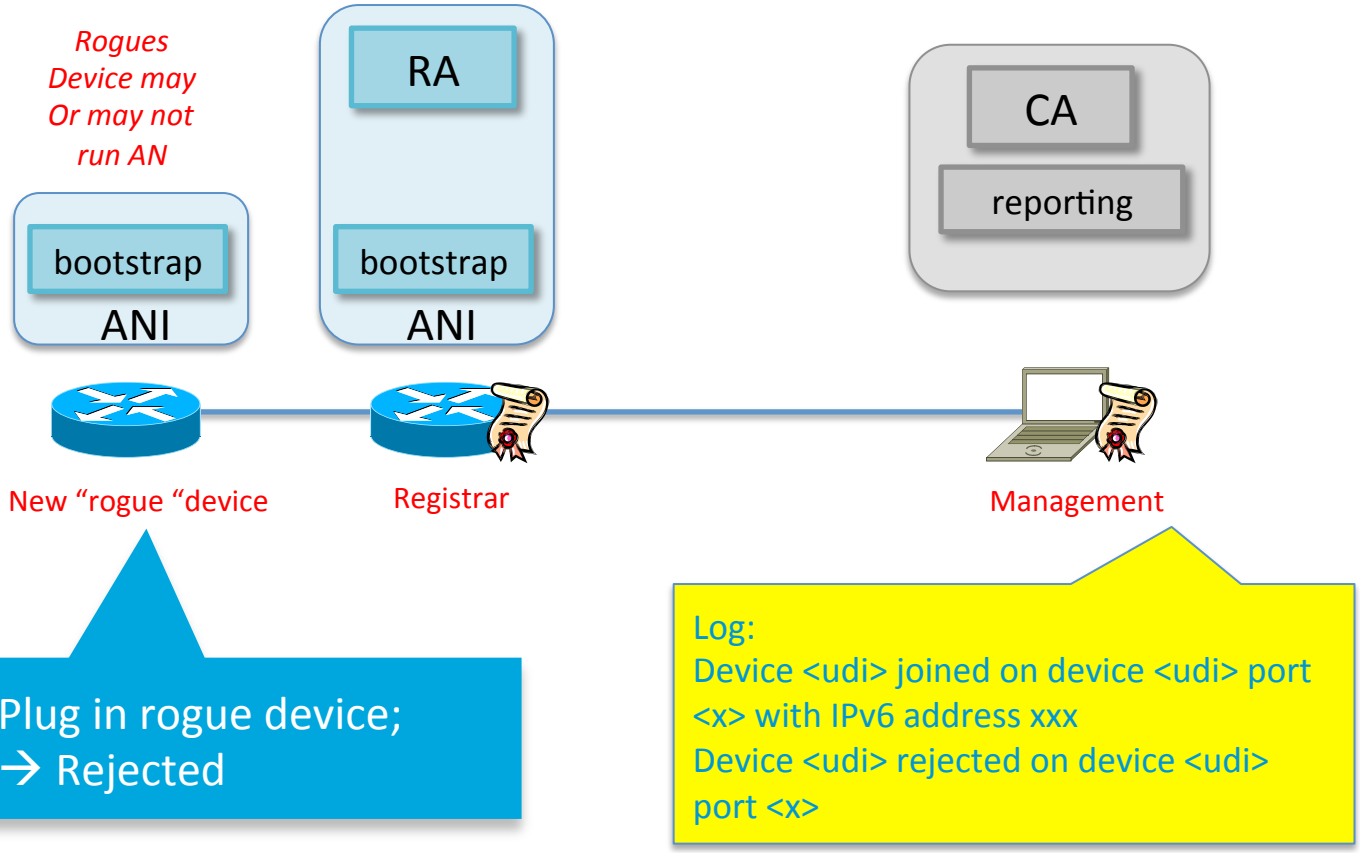
2) Connect a New Device



Notes:

- Complete zero-touch: Device identified by its UDI (later: SUDI)
- If device in white list, no operator intervention at all
- If not, need to manually accept new device based on (S)UDI on registrar

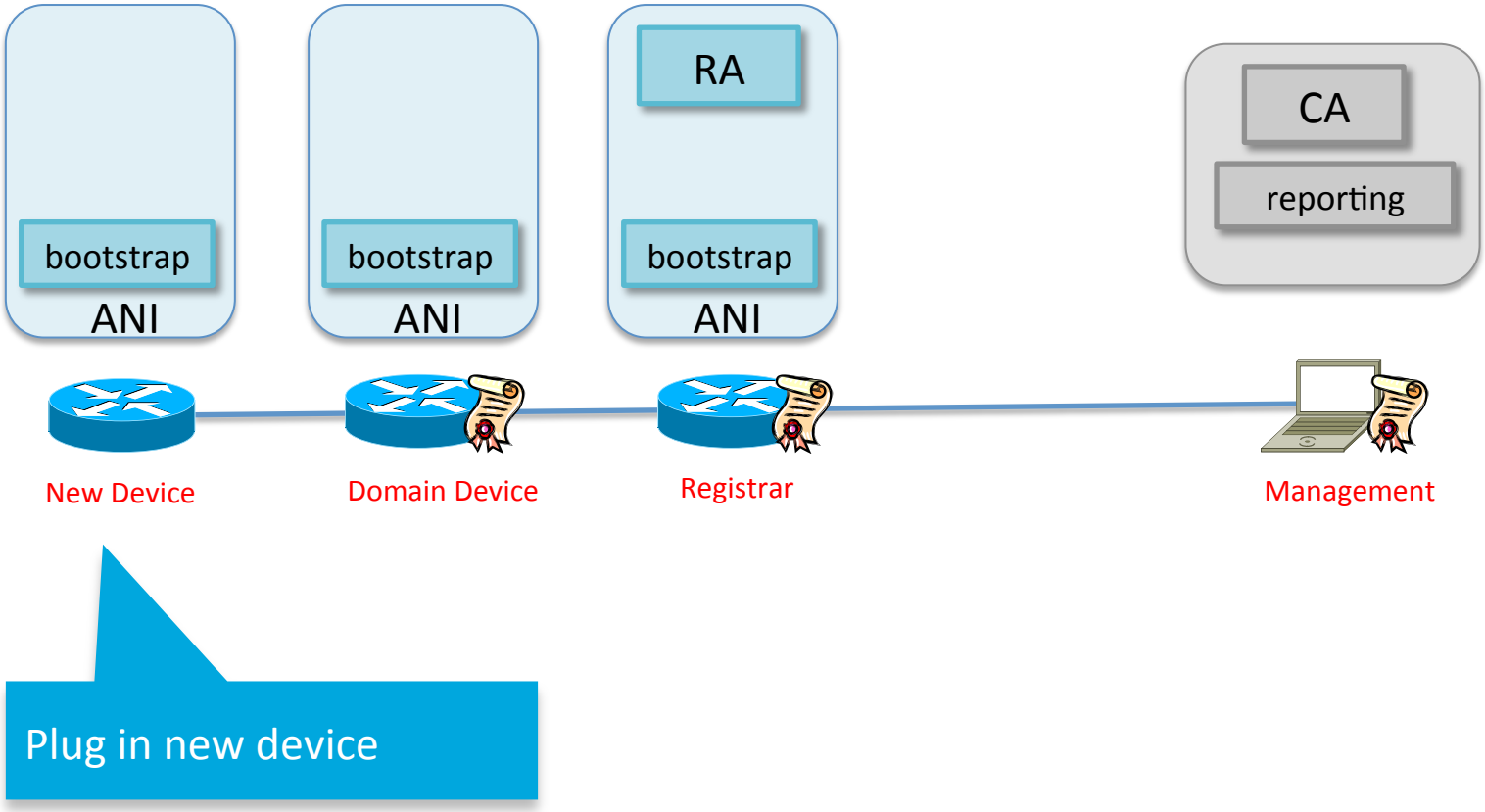
2b) Connect a rogue Device



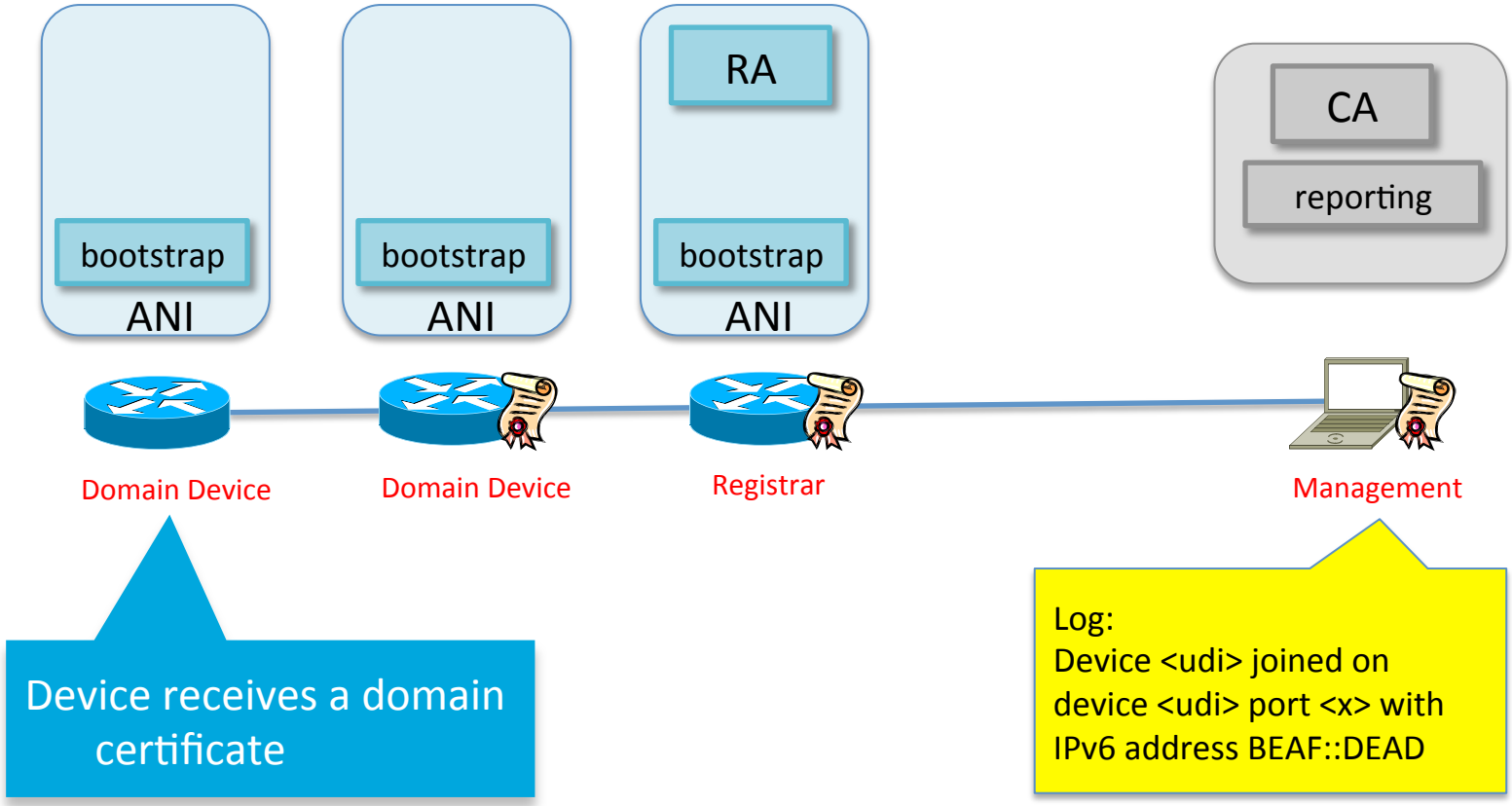
Notes:

- Rogue device means: UDI not known
- Later (with SUDI): Not a Cisco device, or UDI fake

3) Connect the next new Device



3) Connect the next new Device



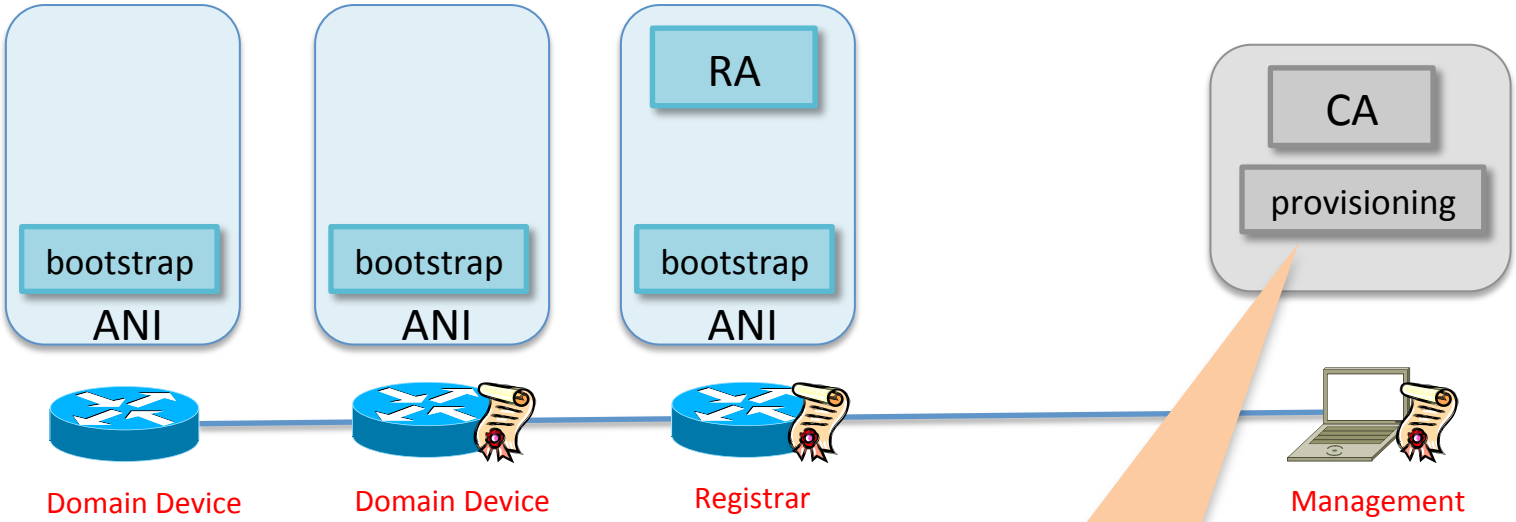
Q: How do we do this ?

A: Traditional way without ACP

When leveraging just AN bootstrap, no ACP implemented.

B: With virtual out-of-band channel - ACP

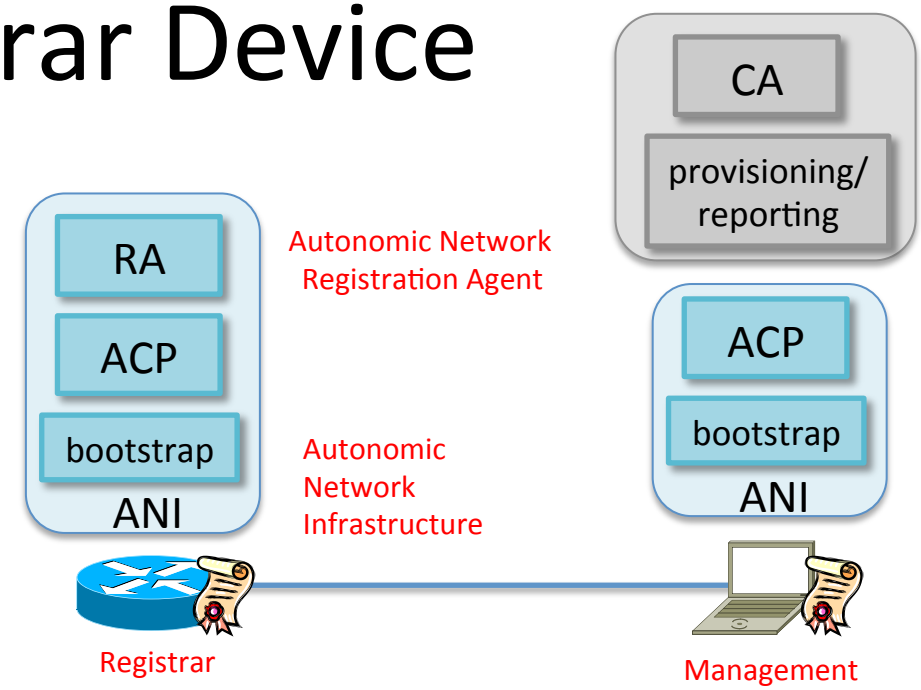
3A) Connect the next new Device



Traditional ("state of the art") approach:

- Provision bootstrap config into all transit devices between proxy and RA
- Bootstrap config domain specific:
  - Routing protocol, addressing policy, security policy
- Problems:
  - Fragile: Domain specific policies can break bootstrap protocol
  - Tighly coupled: enrollment depends on customer specific provisioning system.

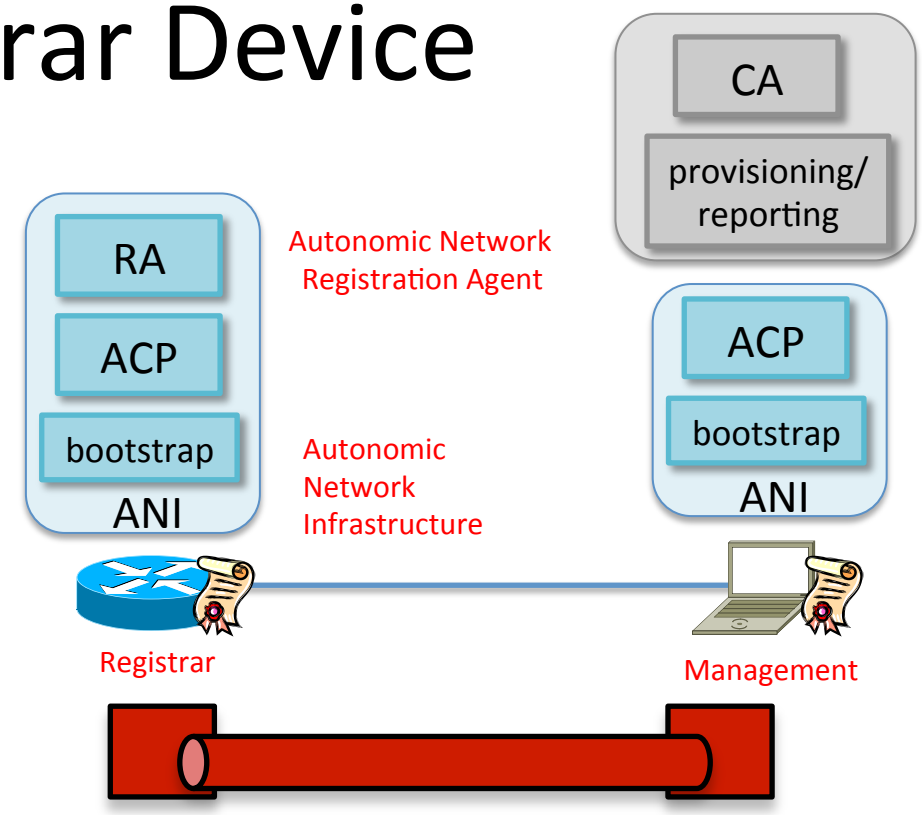
1) Set-up Registrar Device



1. Enable Registrar function
2. Enter domain name
3. RA router enrolls itself into domain, no ACP required.
4. (optional) Upload a white-list of UDIs: new devices are checked against this list

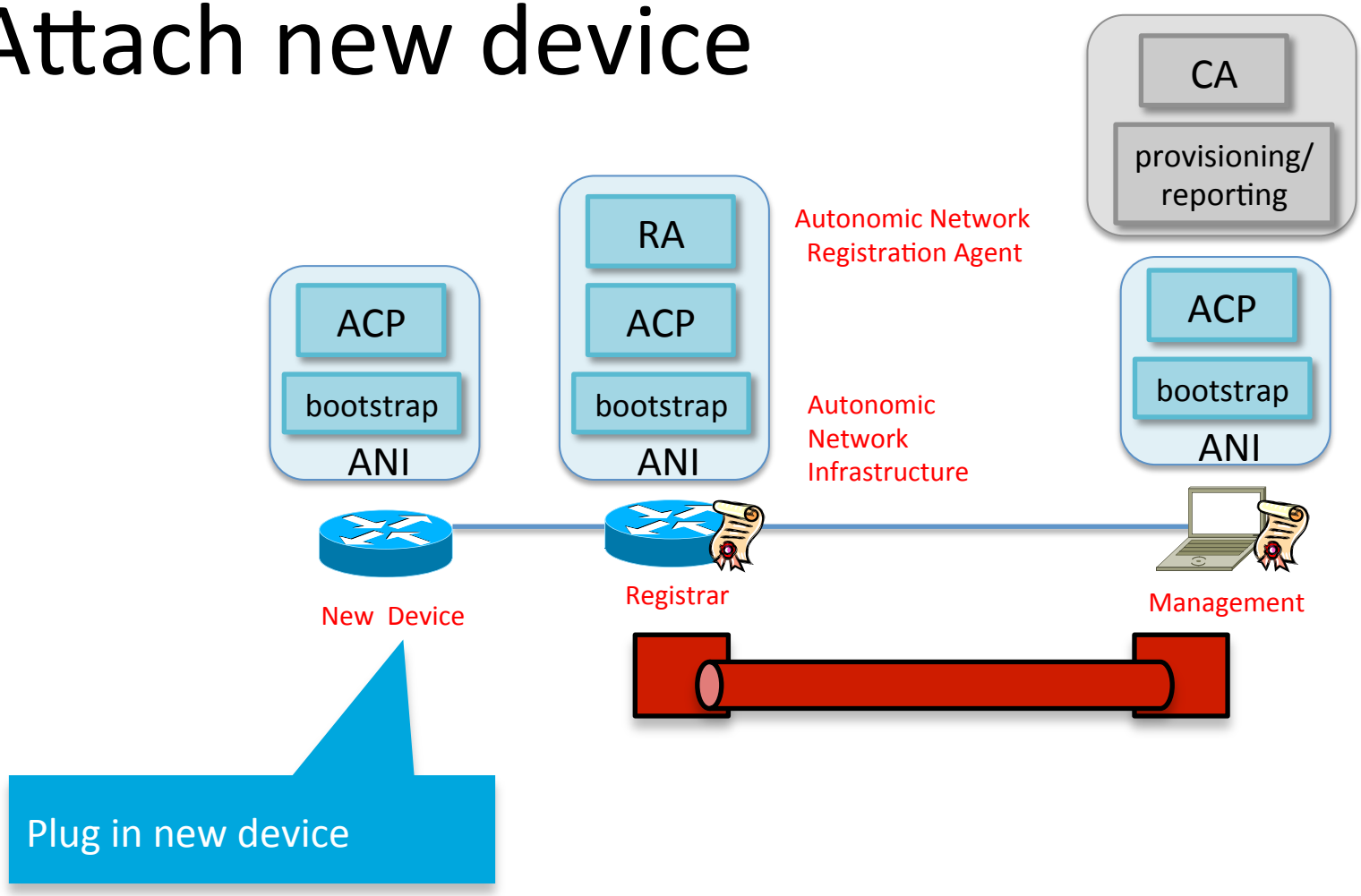
ACP module always running on AN enabled router or management device.

1) Set-up Registrar Device

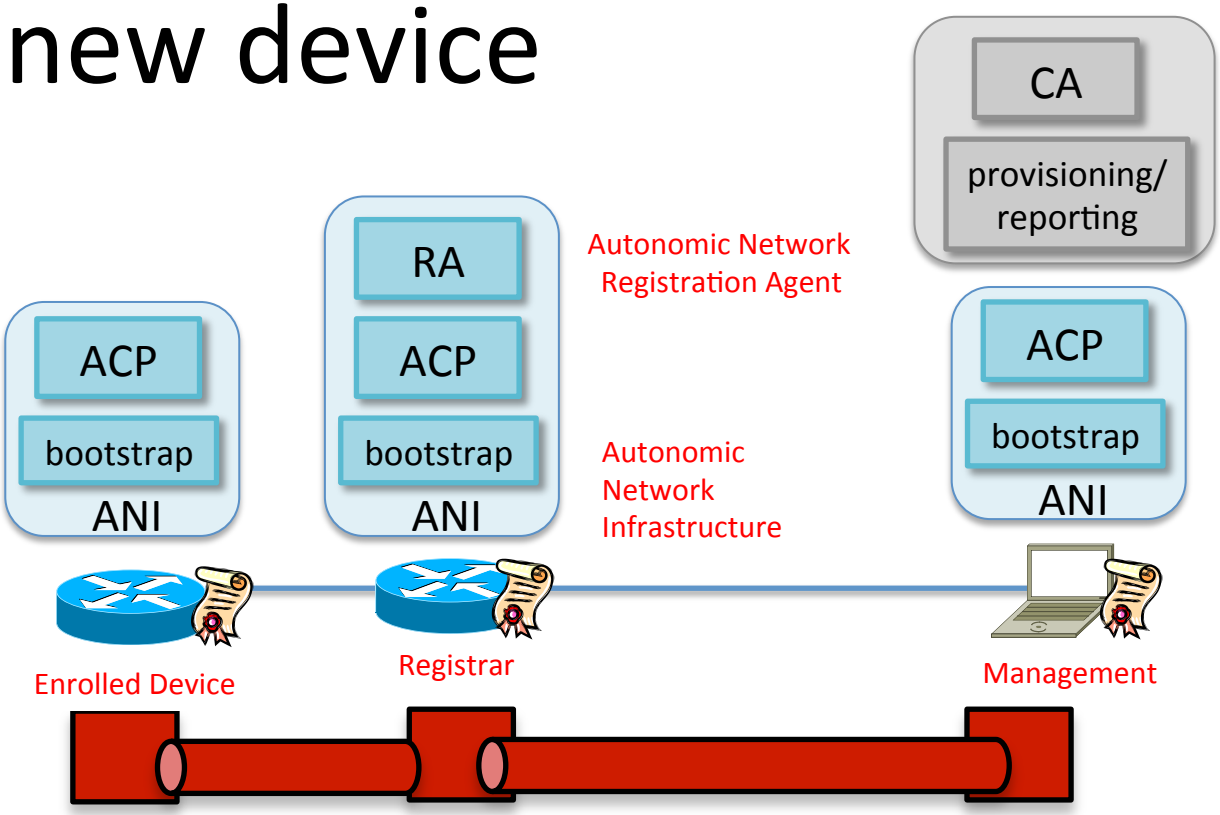


1. ACP on router/AN-management device automatically discovers
2. Secure channel is built between router (registrar) and management system
3. ACP in router is non-configurable = undestroyable
   - Uses fixed routing protocol – RPL
- ACP to management stations not required for any enrollment of devices, but leveraged for managmenet system accessing all enrolled devices (SNMP, SSH, Netconf, PnP – whatever OAM procedure is used).

2) Attach new device



2) Attach new device

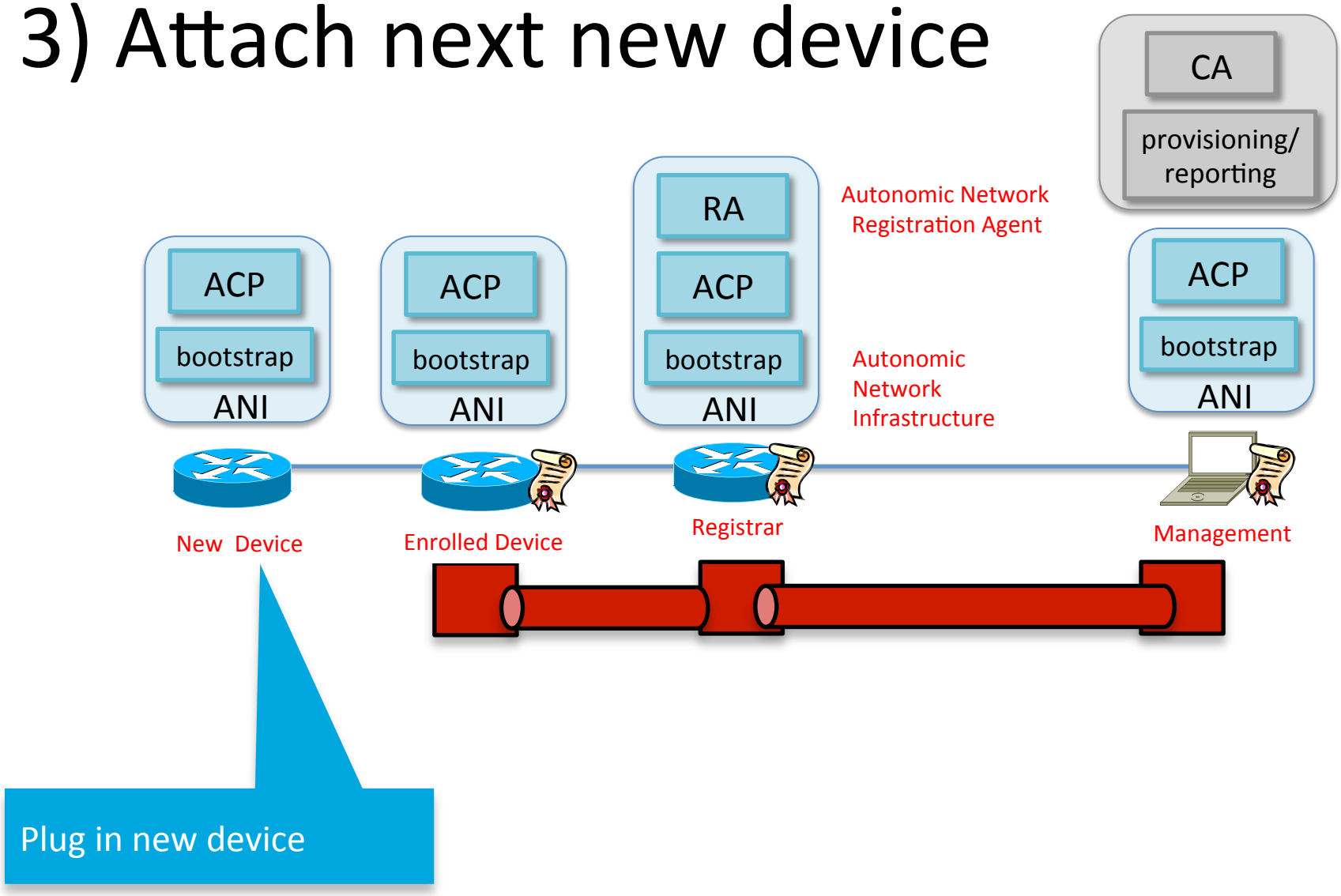


4. Device receives a domain certificate
5. Device builds ACP to neighboring already enrolled router(s)

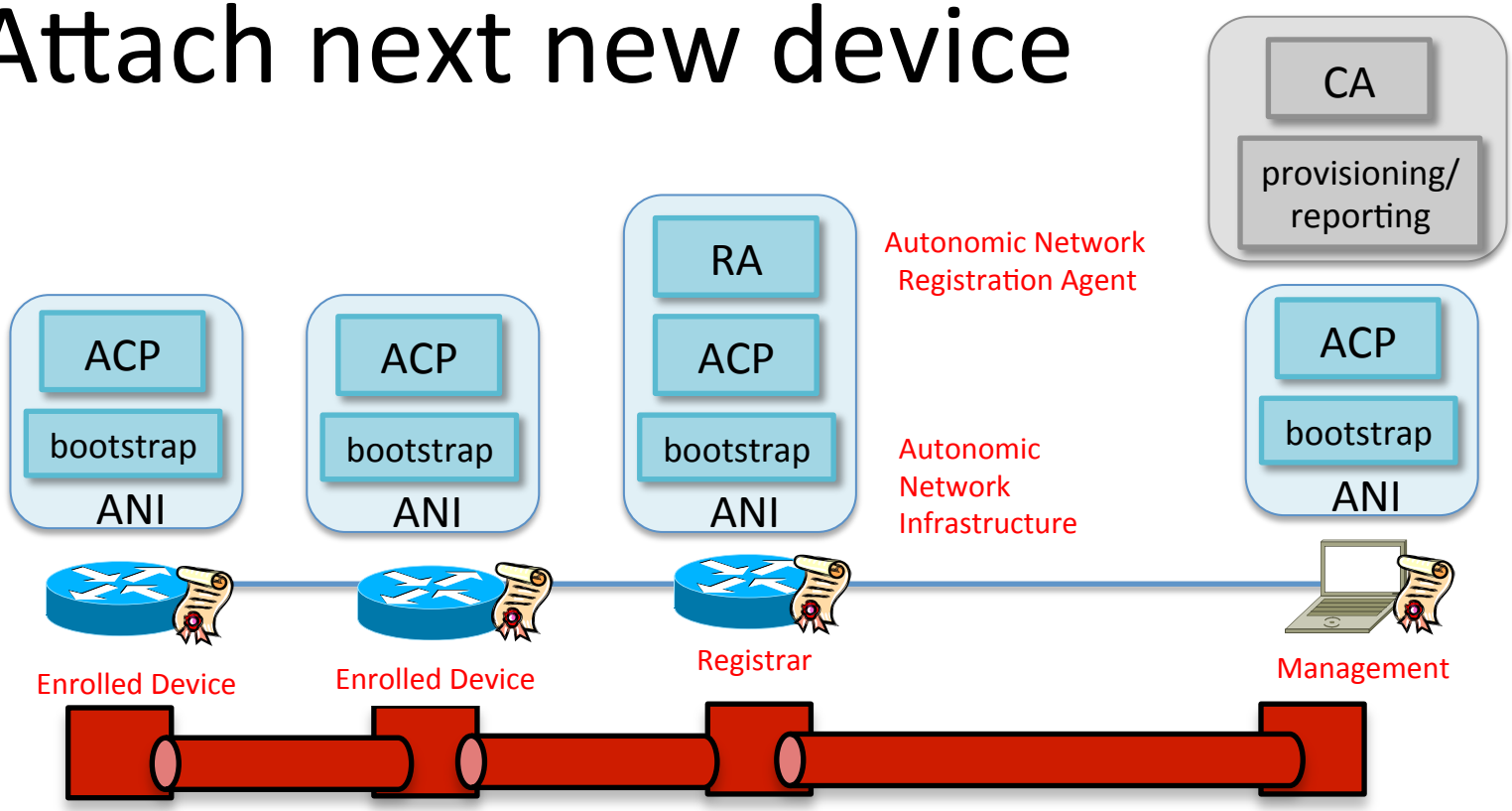
1. Neighboring AN device acts as enrollment proxy for new device.
2. Uses ACP to talk with RA

3. Log:
Device <udi> joined on device <udi> port <x> with IPv6 address DEAD::BEEF

3) Attach next new device



3) Attach next new device



4. Device receives a domain certificate
5. Device builds ACP to neighboring already enrolled router(s)

1. Neighboring AN device acts as proxy for new device
2. Uses ACP to talk with ANRA

3. Log:
Device <udi> joined on device <udi> port <x> with IPv6 address DEAD::BEEF

ACP summary

- No pre-staging config necessary to bring up arbitrarily many devices.
- Full IPv6 reachability from NOC to all enrolled devices
- Greenfield (shown) and brownfield rollout
- Brownfield example: change routing/ addressing/security policies in existing network
  - Breaking connectivity does not impact ACP – its non-configurable

Next IETF ?!

- More than this in drafts and/or prototype or released implementation
  - Protocol details
  - Intent
  - MASA
  - Partial deployment
  - Service discovery/utilization
  - Architecture