CAPPORT Architecture

draft-larose-capport-architecture-01

Authors: K. Larose, D. Dolson

Architecture Updates and Discussion on Working Group Direction

Presenter: David Dolson

Updates

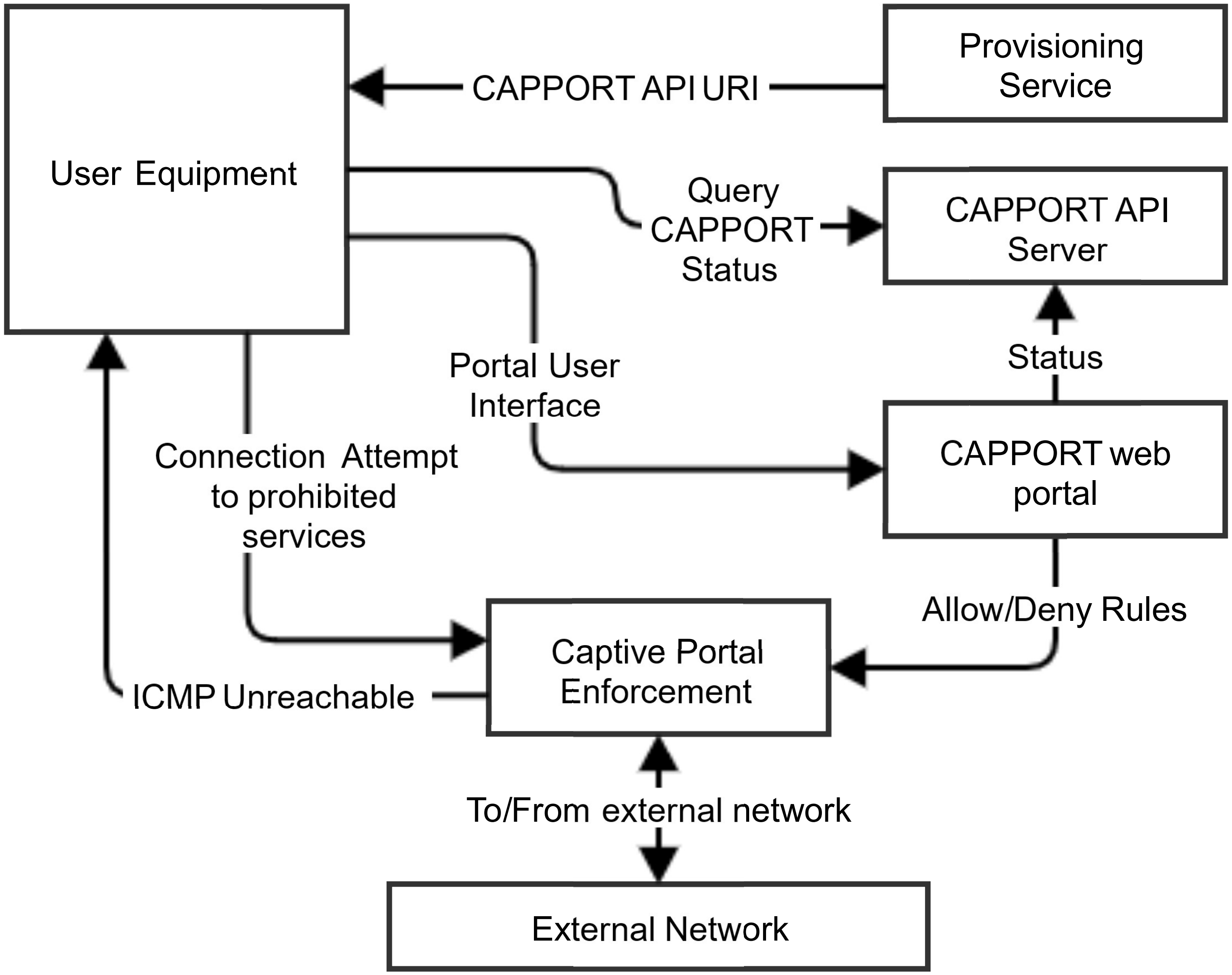
Document version `-01` includes feedback from last meeting and mailing list.

- More emphasis on end-user device having voluntary functions
- Added architectural principles
- New section on Provisioning Domains (PvDs)
- Use of requirements language (RFC 2119)
- Removed IoT device (device without a browser) from scope
- Component Diagram: distinguish API from Web portal
- Attempting to improve precision of language

Principles

- No man-in-middle of DNS, HTTP, etc.
- Use Internet Protocol (vs. access-specific)
- Notify on any protocol, not just port-80 HTTP
- Explicit captivity: machine detectable
- Backwards compabibility, incremental deployment
- Facilitate trust mechanisms

Components and Protocols



Enforcement

- Currently, this function blocks most traffic (except within walled garden) and modifies port-80 HTTP or DNS
- We propose the new Enforcement function sends a form of ICMP "unreachable" message.
- This tangibly improves the reaction to non-port-80-HTTP
- HTTP modification of http port-80 may be used for some time

API

- Expected to be an idempotent RESTful API
- Basic: Read-only
  - Indicates captivity
  - Indicates web interface URL
- Advanced:
  - Remaining bytes/time quota
  - Financial Transacting??
- Tangible improvement over man-in-the-middle HTTP modification
- Note: PvD may be a more general approach to getting the info.

Provisioning Domains

- Latest version of capport-arch discusses PvDs.
  - But PvD draft was updated a day later to exclude CAPPORT...
- PvD mechanism for authentication is interesting
  - Avoid masquerading hot-spots?
- To discuss
  - Is it is too special-case for PvD?
  - Is probing necessary anyhow?
  - Redundant with RFC 7710?

Questions

Can we agree on anything? What advances the state of the art?

If the WG adopts this document, what belongs in it?

- CAPPORT ICMP?
- Should there be an API? Read-only?
- Provisioning requirements?
- Discussion about trust & authentication?
- Keep web interface out of scope?
- Keep IoT out of scope?

Trust

Existing mechanisms:

- There is good reason not to trust a URL forced at you over the internet, which leads to browser "sandboxing"
- Which leads to attempts to defeat captive portal detection

But what if?

- What if there were strong authentication mechanisms?
- E.g., You knew you were connected to Hilton Prague portal?
    - Safe to enter name and room number

Problem Statement?

- We reference draft-nottingham-capport-problem-01 , which expired
- Does WG want to adopt the problem statement, or include some of that text here?
- The architecture document does not address all of the problems...