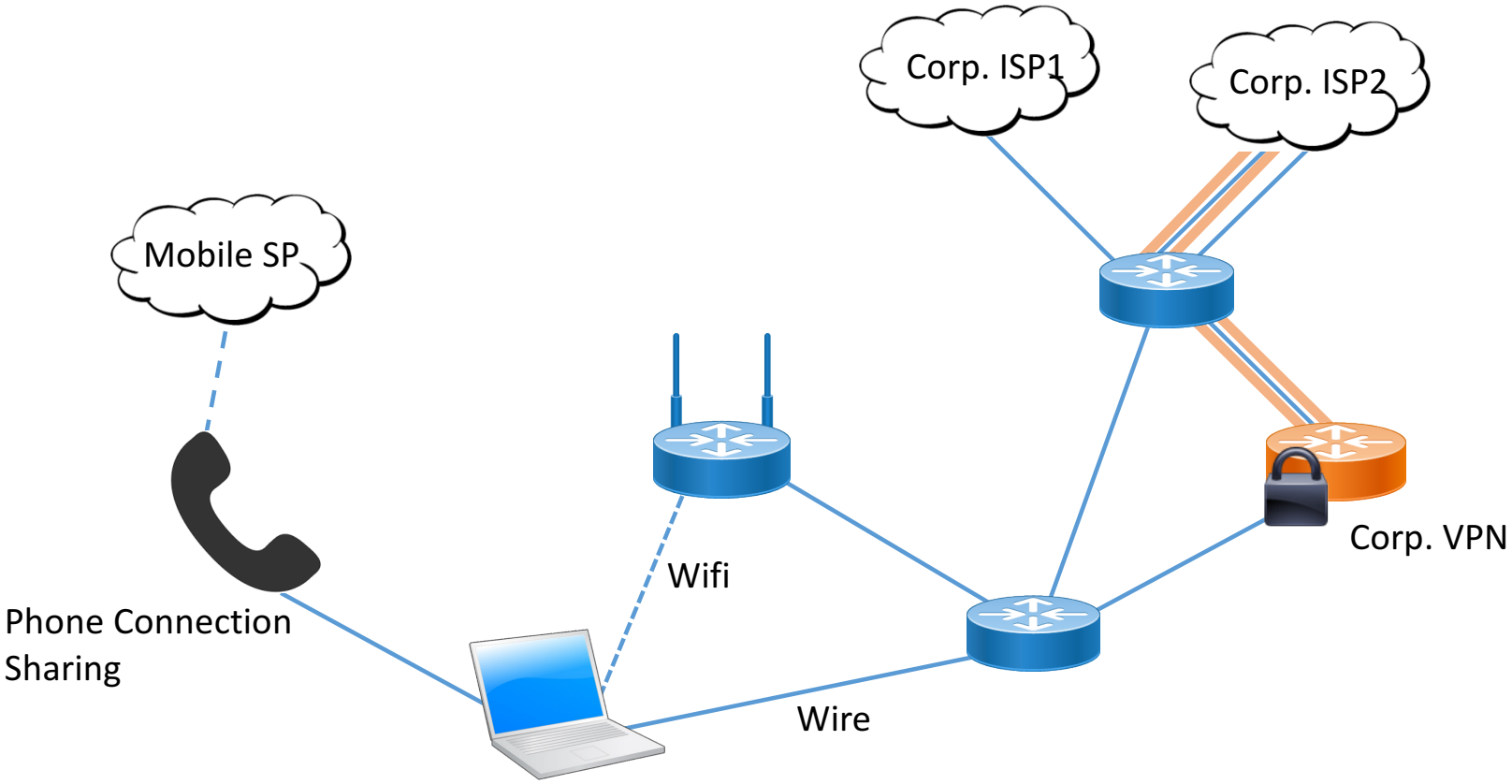
Discovering Provisioning Domain Names and Data

draft-bruneau-intarea-provisioning-domains-01

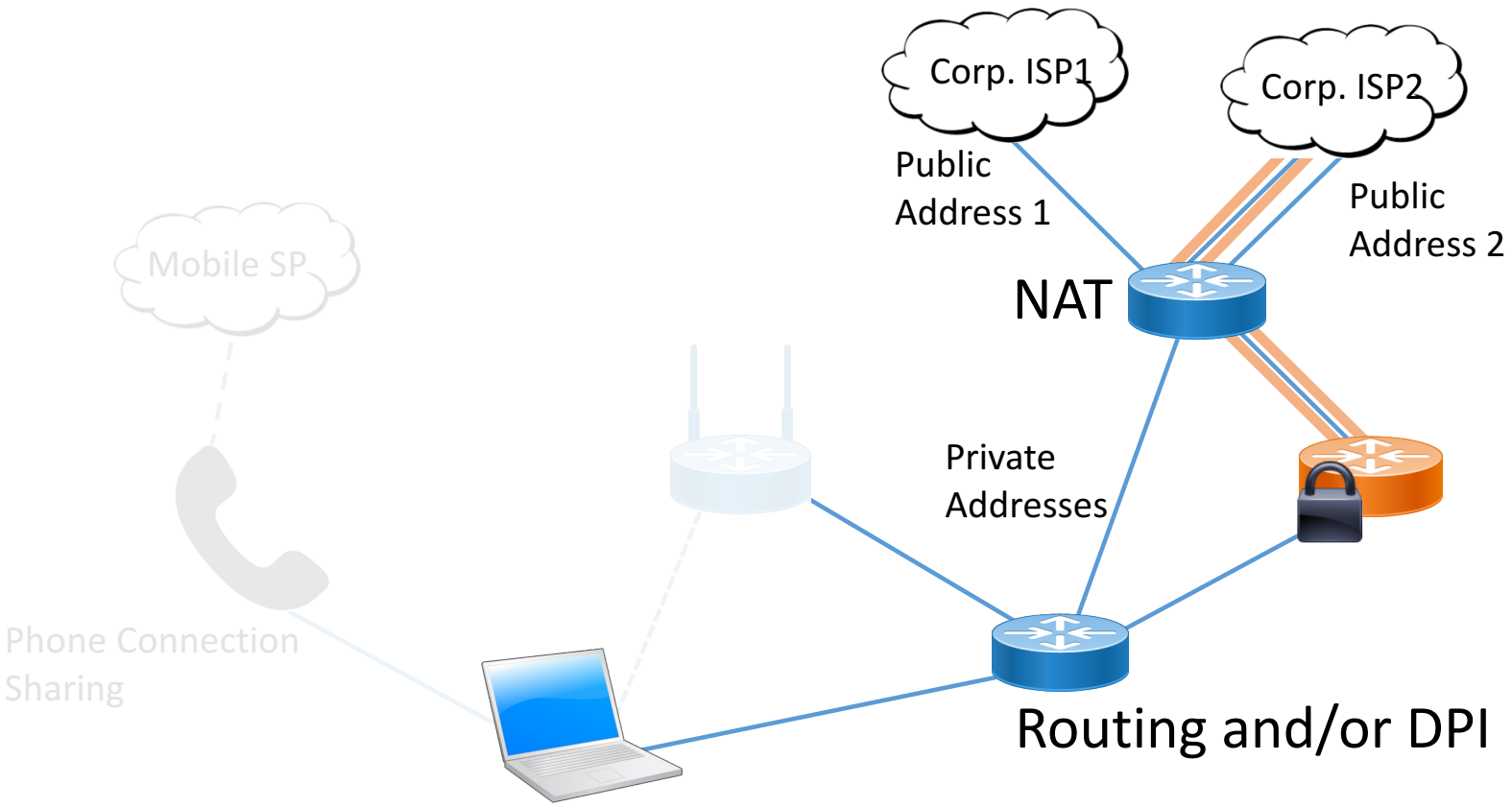
B. Bruneau, P. Pfister, D. Schinazi, T. Pauly, **E. Vyncke**

Hosts and networks are multi-homed

Just a few examples…



Multi-Homing, the legacy way…

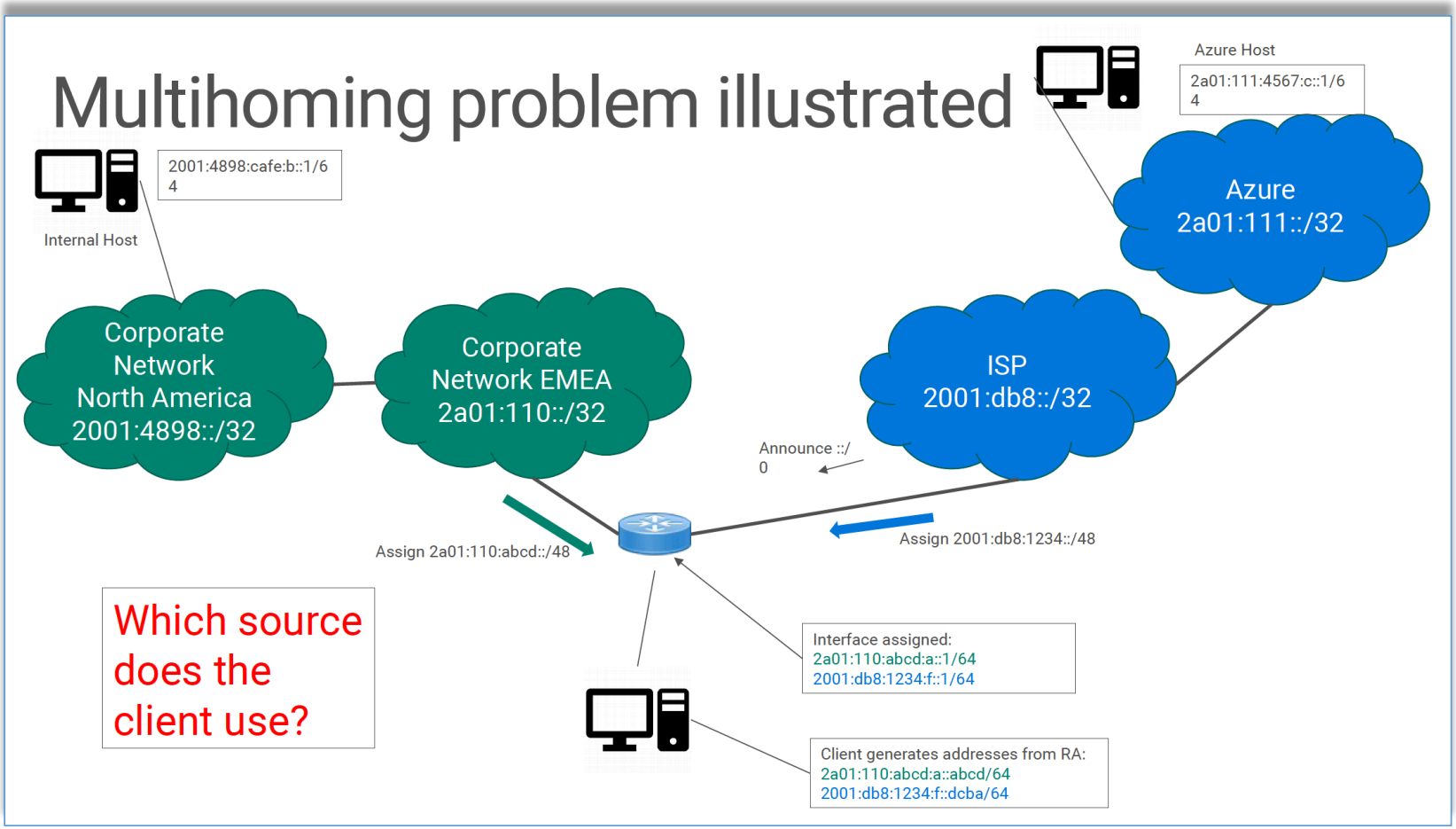


Multi-Homed networks in IPv6

- Assign provider assigned (PA) addresses to hosts.
  - Native to IPv6 hosts (RFC4861, ...)
  - HNCP for home networks (RFC7788)
  - draft-ietf-rtgwg-enterprise-pa-multihoming-01 for corp. networks.

- Teach the hosts to pick and use multiple addresses.
  - IPv6 source address selection (RFC6724)
  - draft-linkova-6man-default-addr-selection-update-00
  - Multi-Path TCP (RFC6824)

- Give the host meaningful information about the addresses.



From Marcus Kean, Microsoft IT, at V6OPS IETF-99

Bundling IP address & DNS resolver

Multihoming and CDNs

- Name lookups for resources stored on CDNs give different answers depending on the network connection
- Host on homenet may look up name using resolver from provider A, then connect to CDN using provider B
- This will generate support requests
- What to do?

Ted Lemon, Homenet WG, IETF-99

Alternative to Bind(socket, [::]:<port>) ?

- In theory, developers could
  - Enumerate all the addresses available on all interface
  - Pick the ones that fits the application's profile
  - Bind individual sockets to each selected address
- In practice, few developers do that
  - Requires tracking address changes
  - Requires testing address properties
  - Tends to not be portable
- And it may not even be available in "service level" API

The purpose of this draft is to:

1. Identify Provisioning Domains (PvDs).

[RFC7556] *Provisioning Domains (PvDs) are consistent sets of network properties that can be implicit, or advertised explicitly.*

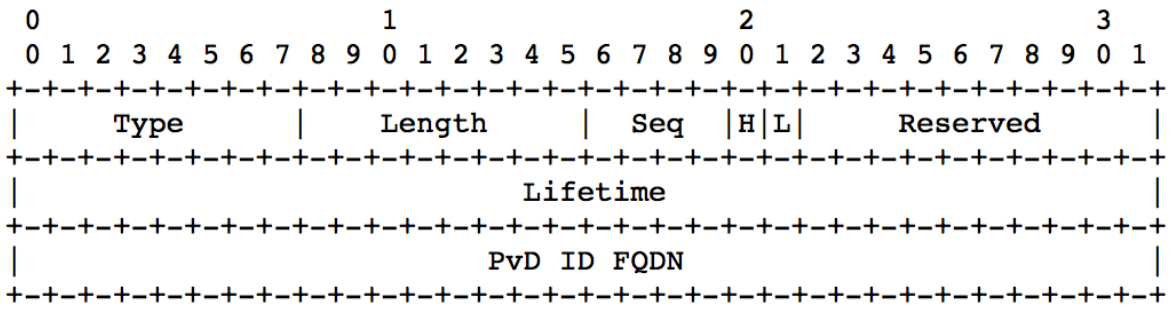
Differentiate provisioning domains by using FQDN identifiers.

2. Give PvD Additional Information.

Name, characteristics, captive portal, etc…

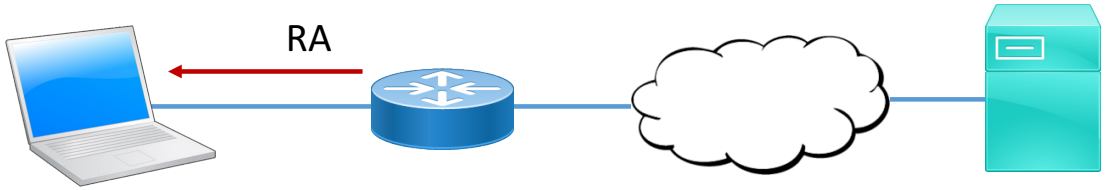
Step 1: Identify PvDs

With the PvD ID Router Advertisement Option



- At most **one occurrence in each RA**.
- **PvD ID is an FQDN** associated with options in the RA**.**
- **Implicit PvDs** (without option) identified by **RA source address and interface**.
- **L bit** to indicate the **PvD has DHCPv4 on the link**.
- **H bit** to indicate **Additional Information is available with HTTPS**.
- Seq. number used for **push-based refresh**.
- Lifetime to indicate PvD ID lifetime.

Step 2: Get the PvD Additional Data

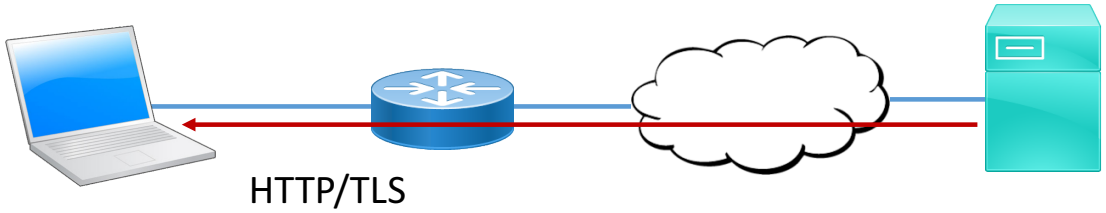


When the H bit is set:

GET https://<pvd-id>/.well-known/pvd (was /pvd.json)

Using network configuration (source address, default route, DNS, etc…) **associated with the received PvD**.

Step 2: Get the PvD Additional Data



When the H bit is set:

GET https://<pvd-id>/.well-known/pvd (was /pvd.json)

Using network configuration (source address, default route, DNS, etc…)
associated with the received PvD.

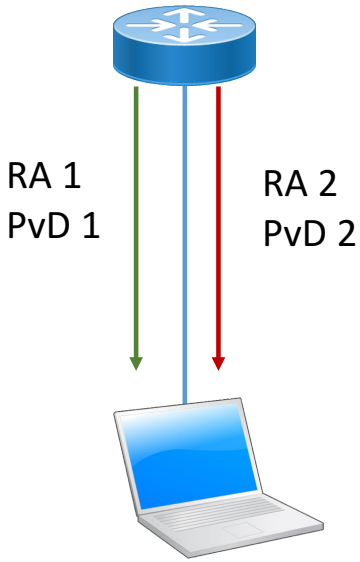
Step 2: Get the PvD Additional Data

```
{
  "name": "Foo Wireless",
  "localizedName": "Foo-France Wifi",
  "expires": "2017-07-23T06:00:00Z",
  "prefixes" : ["2001:db8:1::/48", "2001:db8:4::/48"],
  "characteristics": {
        "maxThroughput": { "down":200000, "up": 50000 },
        "minLatency": { "down": 0.1, "up": 1 }
  }
}
```

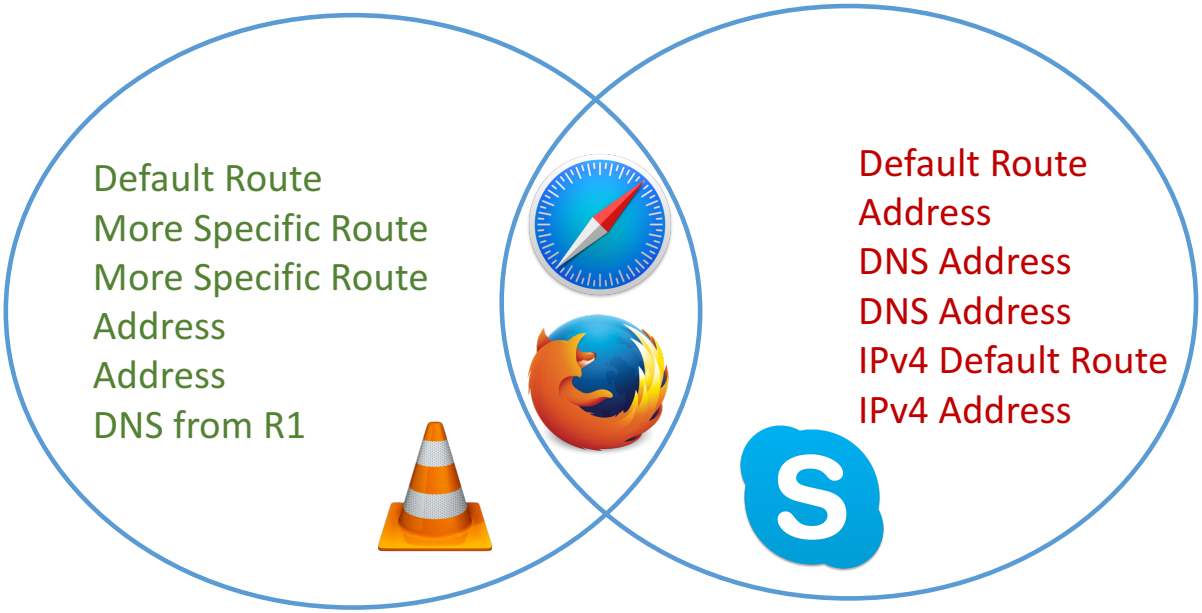
Some other examples (see also https://smart.mpvd.io/.well-known/pvd) :

```
noInternet : true,
metered : true,
captivePortalURL : "https://captive.org/foo.html”
```

Step 3: Host behavior



Hosts and applications **behave according to existing specs** on one or more PvDs.



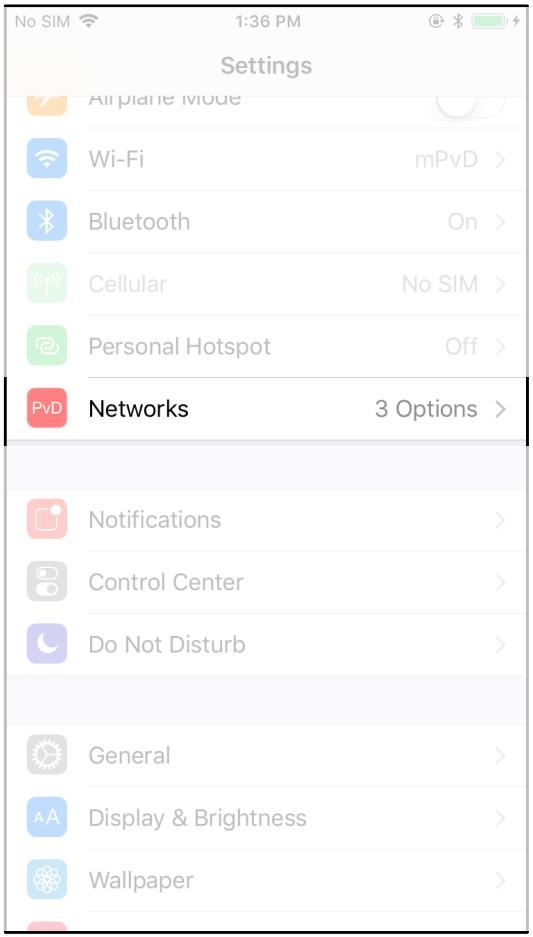
Implementation status

Linux - https://github.com/IPv6-mPvD

- pvdd: A Daemon to manage PvD IDs and Additional Data
- Linux Kernel patch for RA processing
- iproute tool patch to display PvD IDs
- Wireshark dissector

During the IETF Hackathon

- OpenWrt support (daemon and GUI)
- iOS support (Captive portal detection)
- NEAT project integration (Tom Jones)



Implementation status

Linux - https://github.com/IPv6-mPvD

- pvdd: A Daemon to manage PvD IDs and Additional Data
- Linux Kernel patch for RA processing
- iproute tool patch to display PvD IDs
- Wireshark dissector

During the IETF Hackathon

- OpenWrt support (daemon and GUI)
- iOS support (Captive portal detection)
- NEAT project integration (Tom Jones)

See it in action at bits-&-bites this evening

Next steps

- Update the draft based on feedback & hackathon
  - Format of PvD ID as plain ASCII
  - Use of well-known URL RFC5785 rather than pvd.json
- Feedback, suggestions, … are welcome
- Become a working group document ?
- BoF at IETF-100 (WG forming) ?
  - Working PoC implementations in various OS
  - Huge interest of using PvD in V6OPS, 6MAN, CAPPORT, Homenet, … WG