IETF Softwire Interim Meeting

Beijing, Sept. 26th-27th, 2011

Note Well

Any submission to the IETF intended by the Contributor for publication as all or part of an IETF Internet-Draft or RFC and any statement made within the context of an IETF activity is considered an "IETF Contribution”. Such statements include oral statements in IETF sessions, as well as written and electronic communications made at any time or place, which are addressed to:The IETF plenary sessionThe IESG, or any member thereof on behalf of the IESGAny IETF mailing list, including the IETF list itself, any working group or design team list, or any other list functioning under IETF auspicesAny IETF working group or portion thereof The IAB or any member thereof on behalf of the IABThe RFC Editor or the Internet-Drafts functionAll IETF Contributions are subject to the rules of RFC 5378 and RFC 3979 (updated by RFC 4879). Statements made outside of an IETF session, mailing list or other function, that are clearly not intended to be input to an IETF activity, group or function, are not IETF Contributions in the context of this notice.Please consult RFC 5378 and RFC 3979 for details.
A participant in any IETF activity is deemed to accept all IETF rules of process, as documented in Best Current Practices RFCs and IESG Statements.
A participant in any IETF activity acknowledges that written, audio and video records of meetings may be made and may be available to the public.

Meeting Success Criteria

- Success level 1:
  - Understand the “stateless” problem space(<u>s</u>)
    - Define axis & dimensions of the space
    - Chart all candidate technologies on a map on those axis
- Success level 2:
  - Document the various trade-offs
- Success level 3:
  - Can we have a “unifying approach”?
    - Create building blocks
    - Separate what need to be separated

Agenda

- 1. Introduction, problem space
- 2. Deep dive:
  - Give ample time to present & understand all technologies
    - We never had time to do that in the regular IETF meetings
    - **In sake of fairness, all presentations (except for 6rd & multicast) have been accepted**
    - 30 min per presentation
    - Order in which we received the request
- 3. ISP input
- 4. Divide & Conquer
- 5. Conclusion

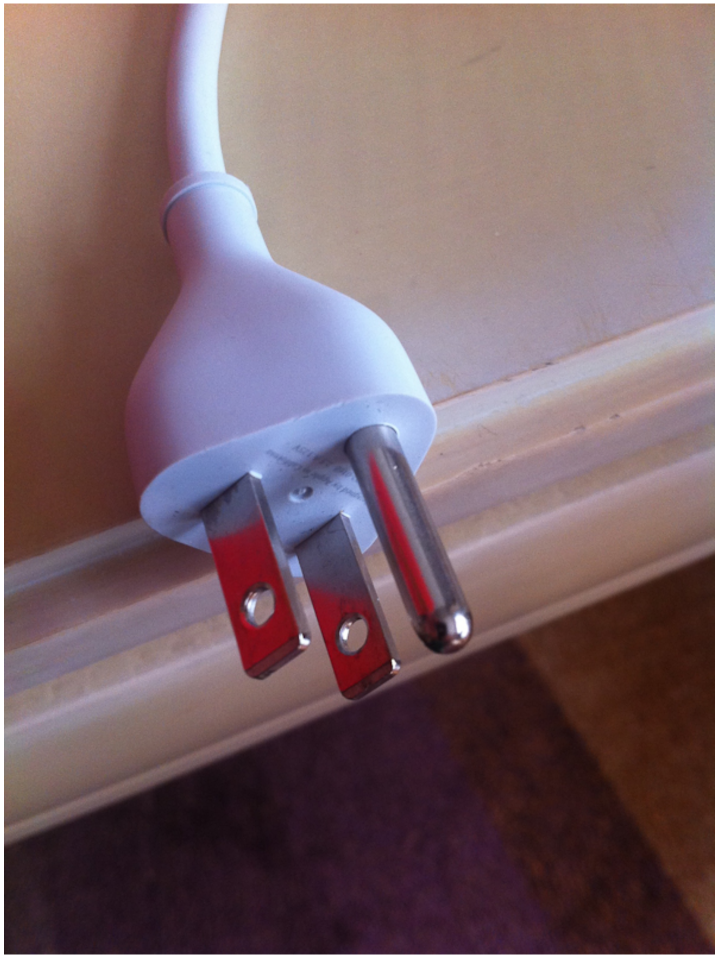
Why do standards matter?

The beauty about standards is that there are more than one to choose from….

How many engineers does it take to plug a laptop into a socket?

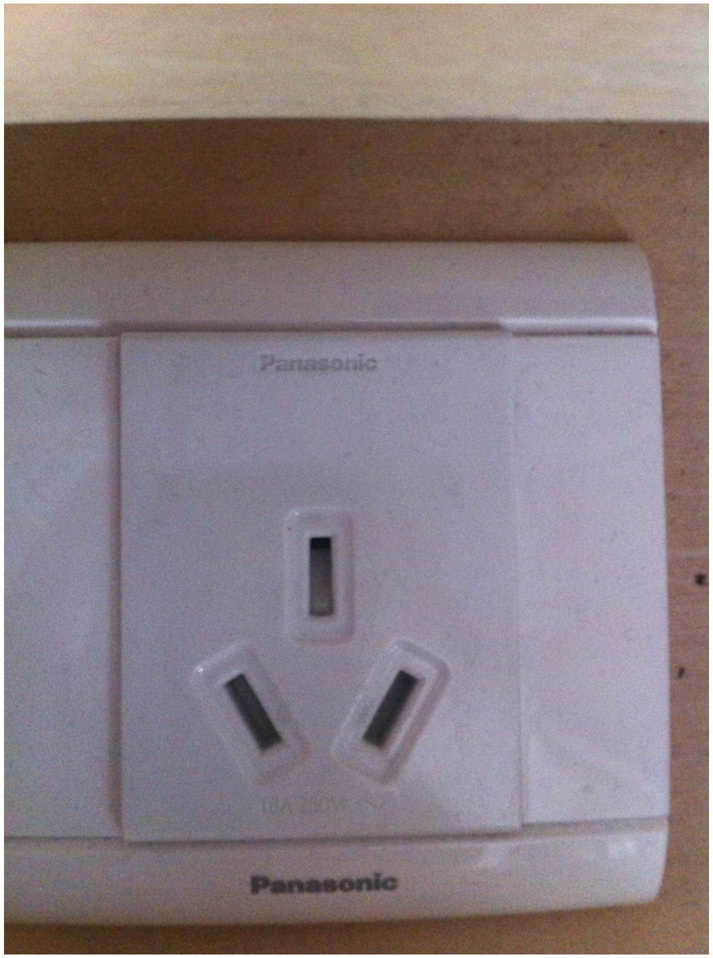
Based on a true story…

1st Attempt



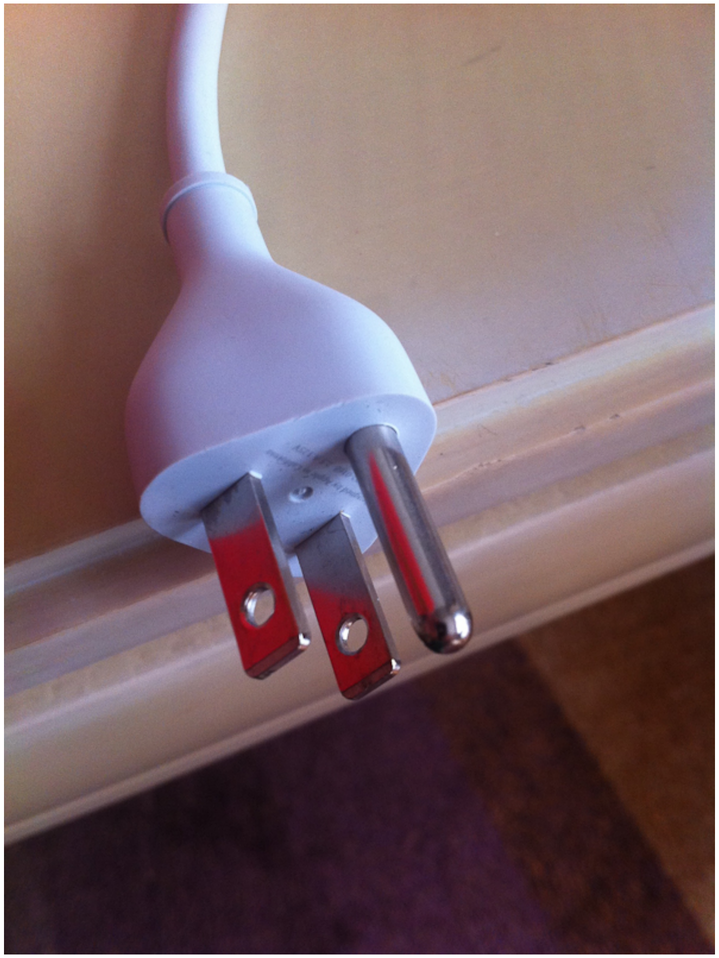
Laptop plug

Wrong shape



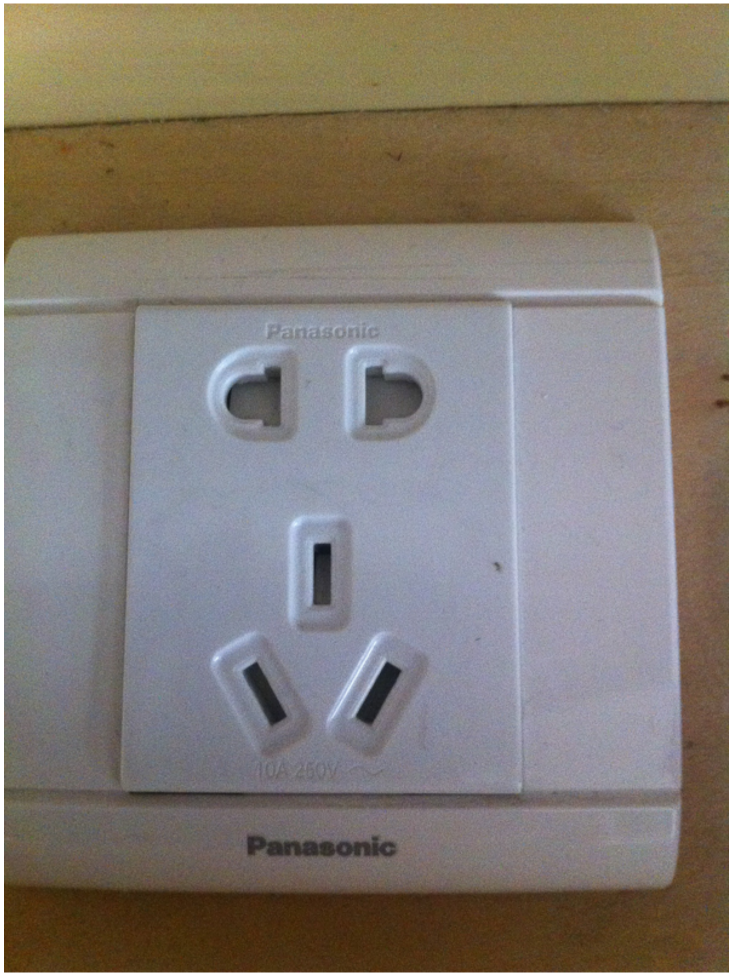
Hotel room socket

2nd Attempt



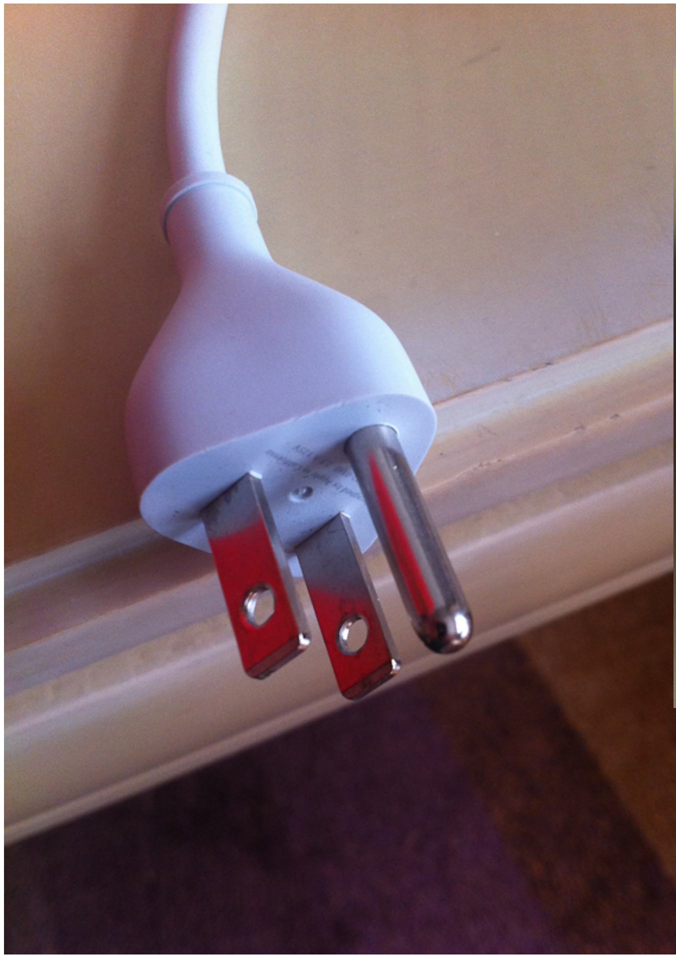
Laptop plug

No ground

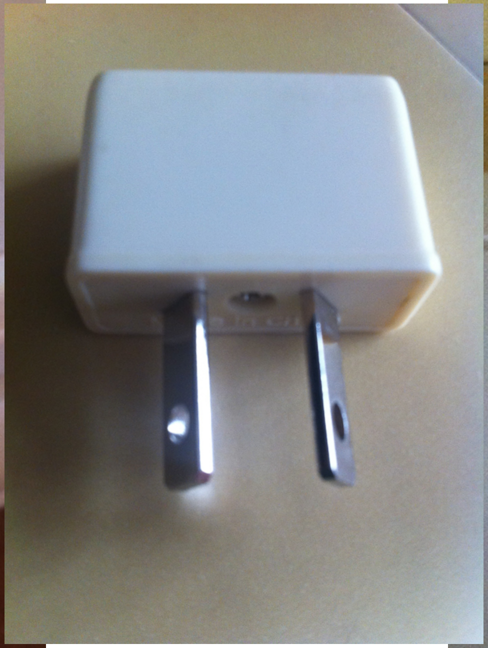


“multi-protocol” socket

3rd Attempt

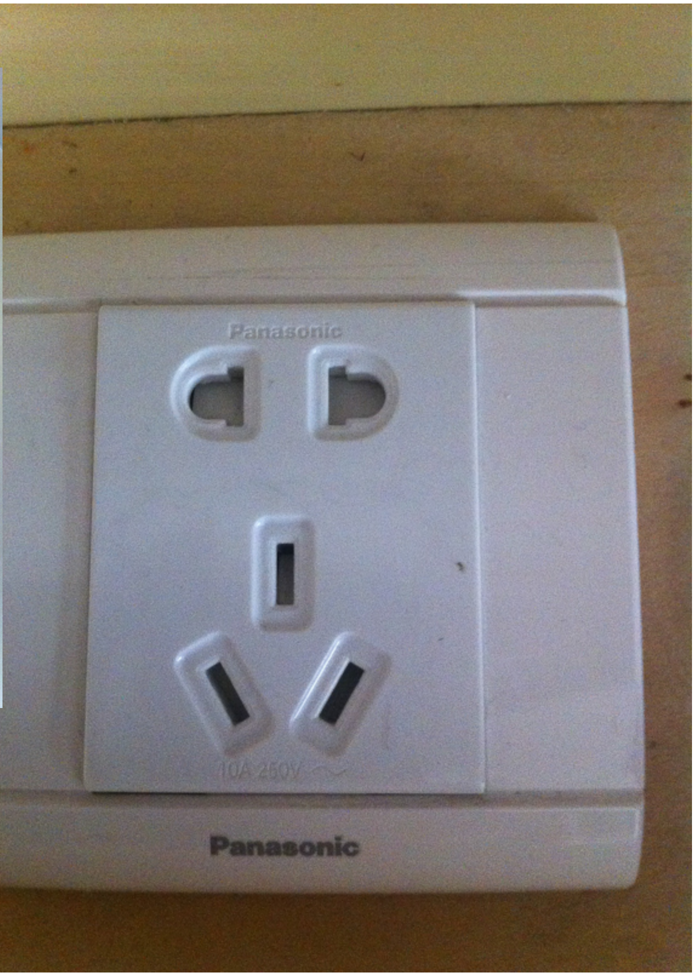


Laptop plug



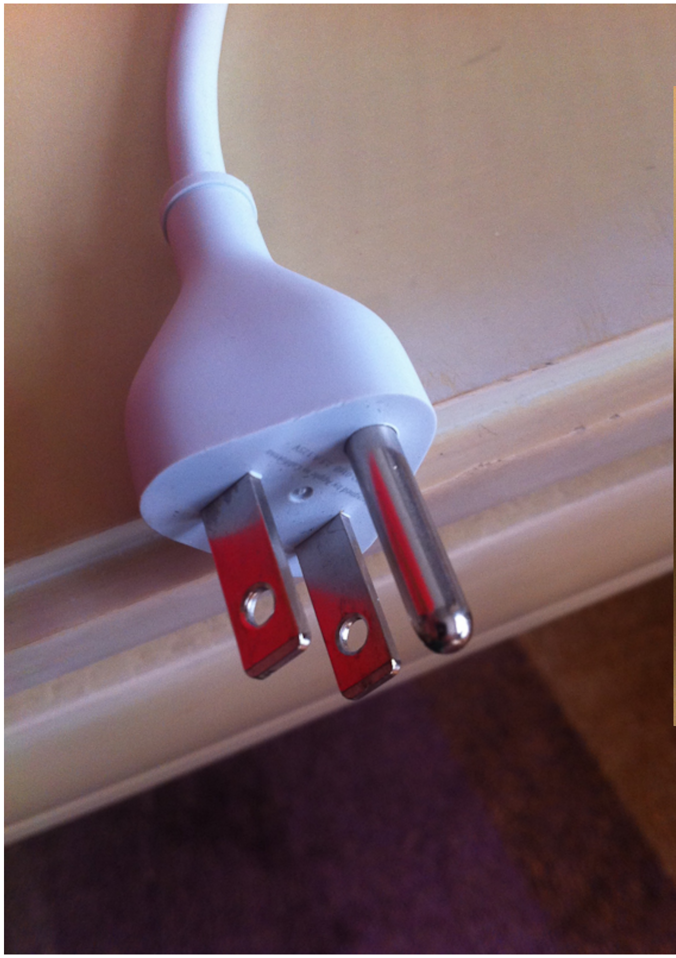
AU adaptor

Ground safety prong missing

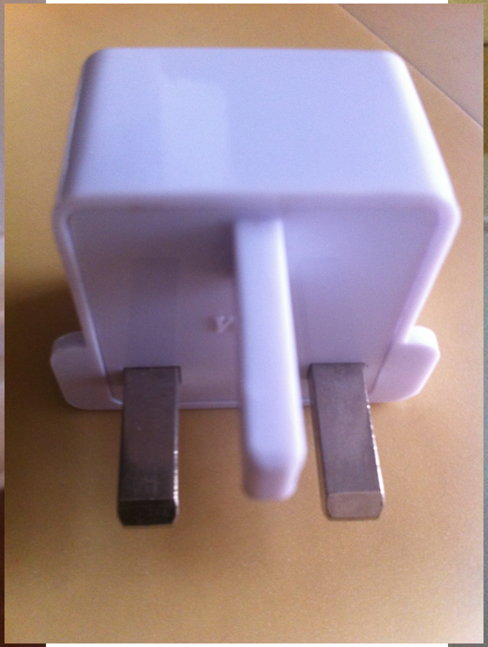


“multi-protocol” socket

4th Attempt

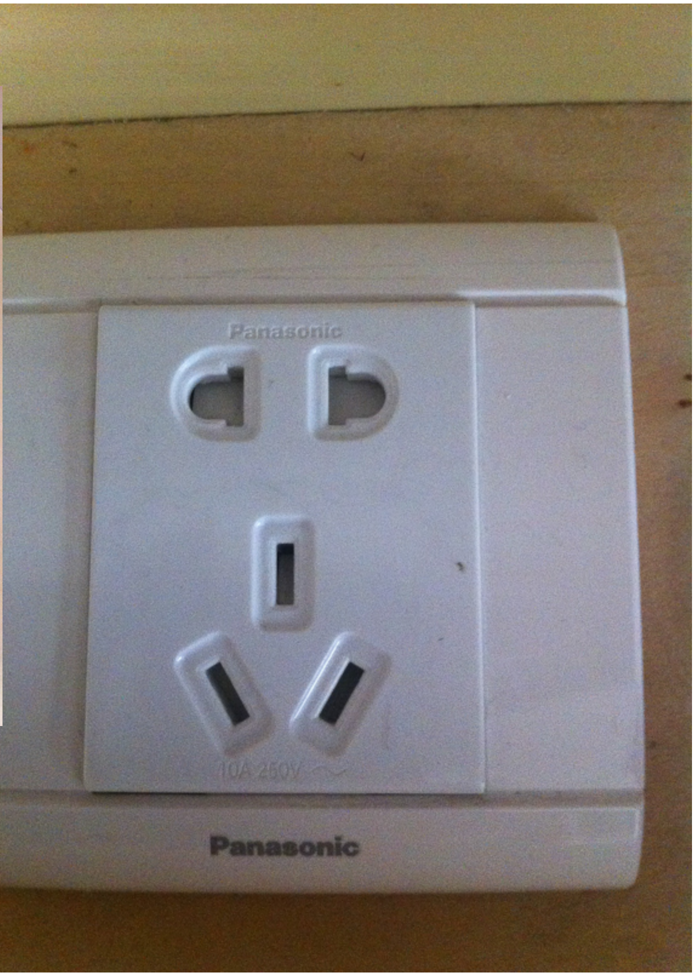


Laptop plug



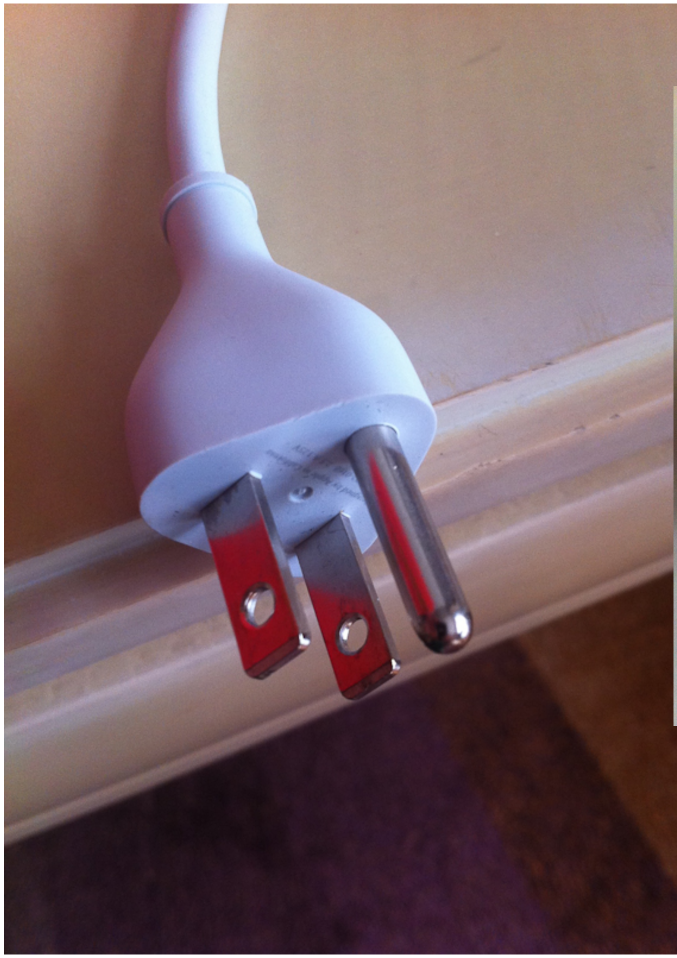
UK adaptor

Wrong shape

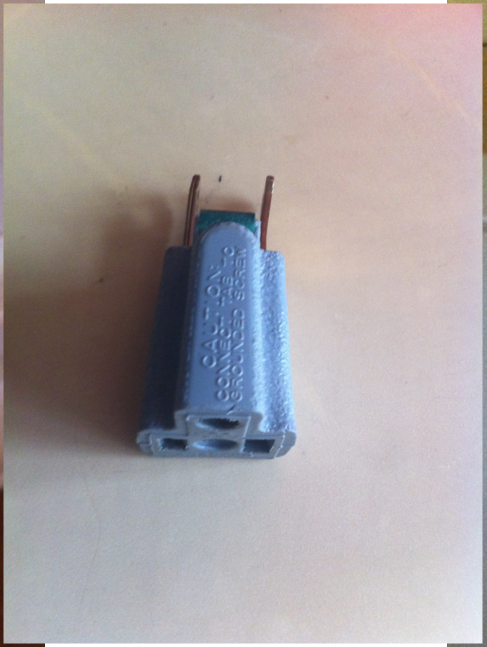


“multi-protocol” socket

5th Attempt

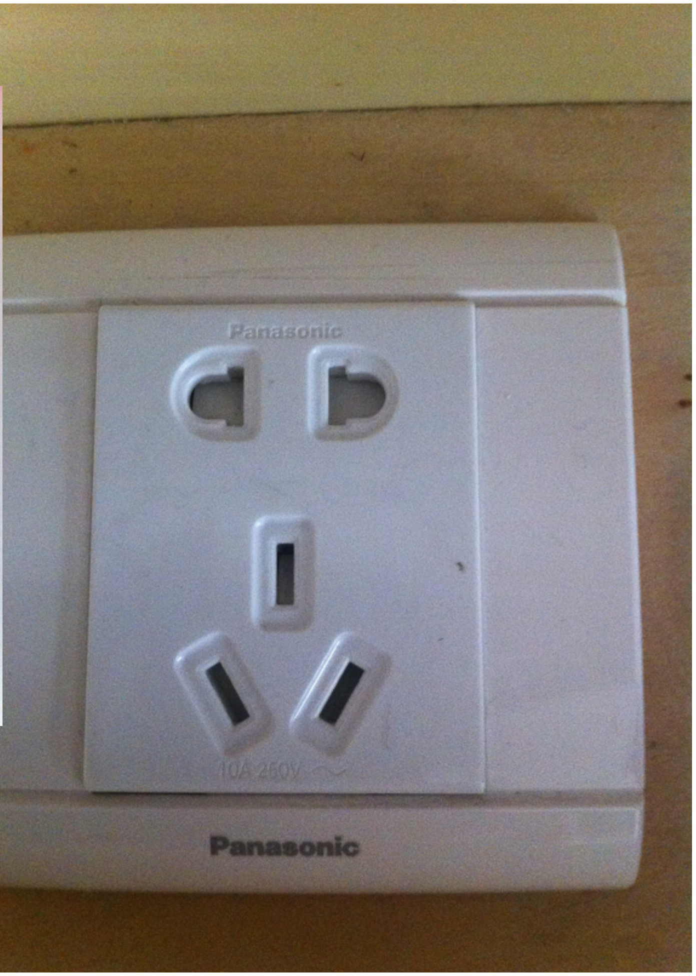


Laptop plug



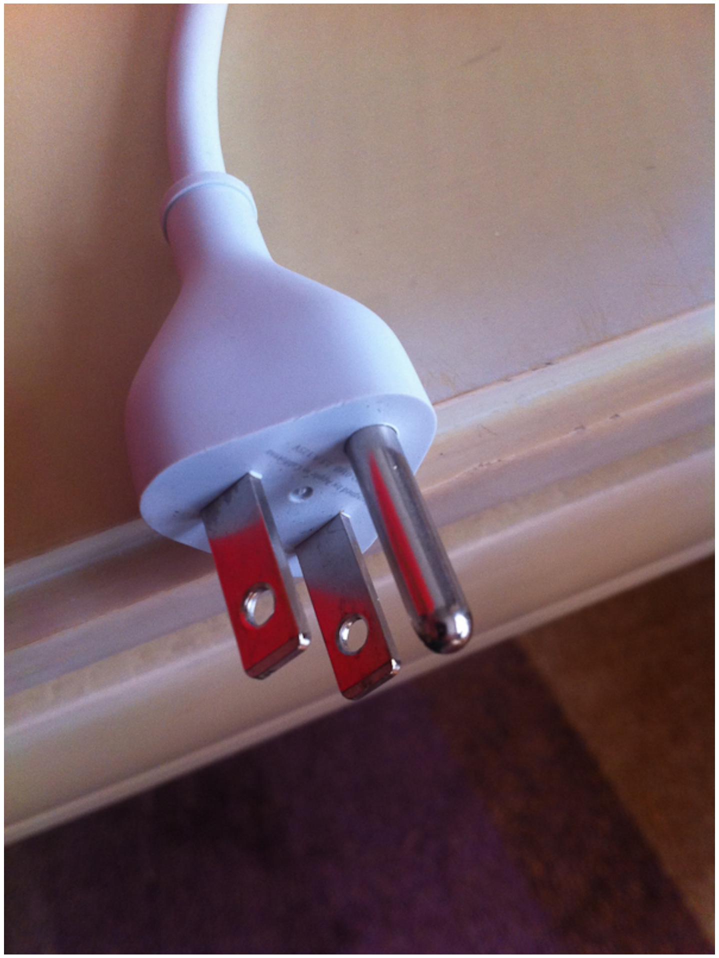
US adaptor

"Hot" prong too wide



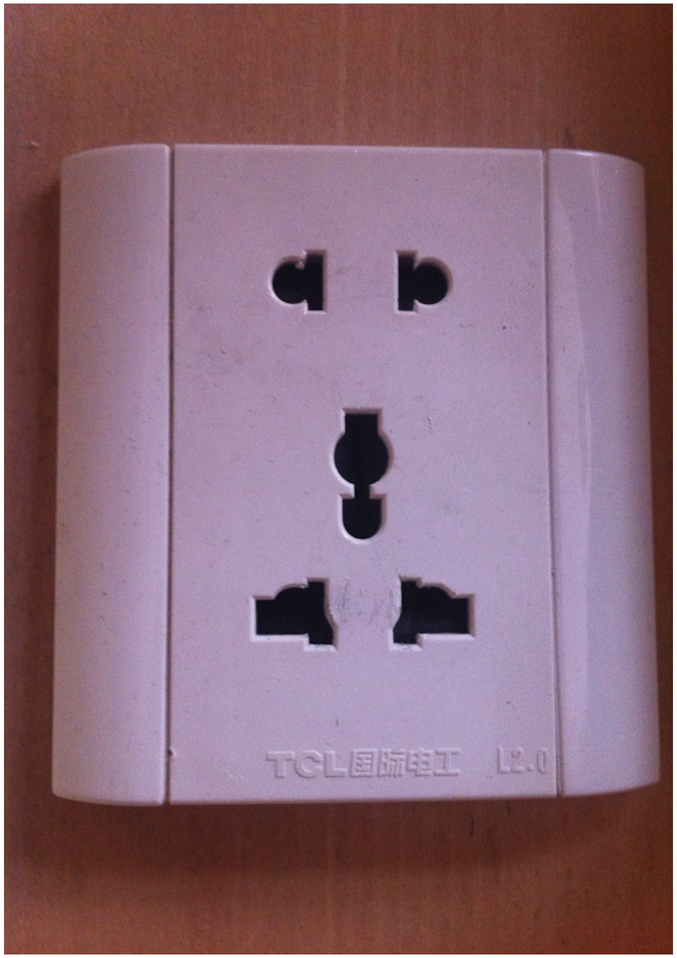
"multi-protocol" socket

6th Attempt



Laptop plug

Victory!



“multi-protocol” socket v2.0

LET’S NOT DO THAT AGAIN…

“Stateless” Problem Space

What we are not talking about

- Connecting IPv6 islands
  - IPv6/IPv4 encapsulation
  - Translation 6->4->6
- Stateful CGN
  - Stateful NAT64/NAT46
  - Stateful NAT444, DS-Lite

NAT Binding Location

- NAT bindings are maintained on CPE, not CGN

- This part seem to be agreed

Mesh vs. Hub & Spokes

- Hub & Spokes (aka managed service)
  - Traffic goes through single exit point
  - Explicit IPv4 subscriber management (AAA, LEA, ACL, QoS,…) done on exit point, independently from IPv6 subscriber management.

- Mesh (aka unmanaged service)
  - Direct shortcuts between internal islands
  - Multiple exits to outside world
  - Implicit IPv4 subscriber management derived from IPv6 subscriber management (AAA, LEA, ACL, QoS,…)

Translation vs Encapsulation

- Terminology
  - **Tunnels**: usually point-to-point, set-up protocol
  - **Encapsulation**: usually point-to-multipoint or multipoint-to-multipoint, no set-up protocol
- This is about “how is my IPv4 packet transported from point A to point B across an IPv6 network.
  - More on this later

How does a CPE “learns” its IPv4 address & port range?

- Implicit from IPv6 address
  - Mapping rules
  - Hints from DHCP
  - Congruent IPv4 & IPv6 address spaces
  - Good for contiguous IPv4 address space

- Explicit
  - IPv4 address & port provisioned by ISP
    - E.g. from DHCPv4/IPv6
  - Non-congruent IPv4 & IPv6 address spaces
  - Good for scattered address IPv4 space

Address Mapping Rules

- How to map an IPv4 address + port set into an IPv6 address
- Many candidates with different properties
  - It is about choosing a bit pattern format.
  - More on this later

Integration with Existing Mechanism

- Is this a brand new mechanism?
- Is this the mirror-image of an existing mechanism?
- Is this an extension of an existing mechanism?

Axis	Technology X
NAT bindings	
Mesh vs. Hub & Spokes	
Translation vs. Encapsulation	
How does the CPE “learns” its IPv4 address & port range?	
How does the CPE derive the IPv4 address + port set	
What does the CPE put in the IPv6 dst?	
Integration with Existing Mechanism	

Axis	ID-Boucadair-ipv6-port-range
NAT bindings	CPE
Mesh vs. Hub & Spokes	Hub&Spoke, evolves to Mesh
Translation vs. Encapsulation	Encapsulation
How does the CPE “learns” its IPv4 address & port range?	IPv4 address: DCHPv6 option Port range: DHCPv6 option How does PRR knows those IP & port range? Manual config Comment: Would be easier if DHCP were to go over the tunnel so PRR can learn it
Address mapping rules	Well known ISP prefix + address + port range
Integration with Existing Mechanism	Evolution from DS-Lite

Axis	SA46T / SA46TAS
NAT bindings	TBD
Mesh vs. Hub & Spokes	Mesh
Translation vs. Encapsulation	Encapsulation
How does the CPE “learns” its IPv4 address & port range?	DHCP option, TBD
How does the CPE derive the IPv4 address + port set	IPv4 + port in lower 48 bits Plane ID (VPN ID) in lower 64 bits
What does the CPE put in the IPv6 dst?	Same
Integration with Existing Mechanism	Isatap equivalent for IPv6, L3VPN over IPv6

Axis	4rd-a
NAT bindings	CPE
Mesh vs. Hub & Spokes	MESH
Translation vs. Encapsulation	Encapsulation
How does the CPE “learns” its IPv4 address & port range?	From IPv6 prefix + DHCPv6 hints
How does the CPE derive the IPv4 address + port set	Statelessly advertized mapping rules: Port heads with port set ID
What does the CPE put in the IPv6 dst?	Dst in same same domain: same rule External dst: Same rule, IPv6 address of BR 1 mapping rule per IPv4 domain
Integration with Existing Mechanism	Mirror image from 6rd + port set algorithm

Axis	4rd-b2
NAT bindings	CPE
Mesh vs. Hub & Spokes	MESH
Translation vs. Encapsulation	Encapsulation or double translation
How does the CPE "learns" its IPv4 address & port range?	From IPv6 prefix + hints (eg stateless DHCPv6)
How does the CPE derive the IPv4 address + port set	Statelessly advertized mapping rules: Port set ID. Size of v4 port space derived from IPv6 prefix length.
What does the CPE put in the IPv6 dst?	Dst in same same domain: same rule External dst: Same rule, IPv6 address of BR 1 mapping rule per IPv4 domain
Integration with Existing Mechanism	Mirror image from 6rd + port set algorithm

Axis	4via6
NAT bindings	CPE
Mesh vs. Hub & Spokes	both
Translation vs. Encapsulation	translation
How does the CPE “learns” its IPv4 address & port range?	
How does the CPE derive the IPv4 address + port set	
What does the CPE put in the IPv6 dst?	
Integration with Existing Mechanism	

Axis	Stateless 4over6
NAT bindings	CPE
Mesh vs. Hub & Spokes	Hub & Spokes
Translation vs. Encapsulation	Encapsulation
How does the CPE “learns” its IPv4 address & port range?	IPv4 address: DHCPv4, TR69,… Port range: new DHCPv4 option, TR69,…
How does the CPE derive the IPv4 address + port set	ID.xli-behave-divi-pd No mapping rules on CPE
What does the CPE put in the IPv6 dst?	AFTR IPv6 address
Integration with Existing Mechanism	Derivative from DS-Lite

Axis	dIVI
NAT bindings	
Mesh vs. Hub & Spokes	
Translation vs. Encapsulation	
How does the CPE “learns” its IPv4 address & port range?	
How does the CPE derive the IPv4 address + port set	
What does the CPE put in the IPv6 dst?	
Integration with Existing Mechanism	