— CBOR: packed
— Errata report handling, CDDL tag syntax
— merge prefix and suffix tables
— use second table beyond **affix**: **argument**
— explain **affix** processing of tags ("function tags")
Example: merged argument table

["https://packed.example/foo.html",
 "https://packed.example/bar.html",
 "https://packed.example/ant.cbor"]

51([[],
 ["http://packed.example/"],
 [".html"],
 [pr0(su0("foo")),
  pr0(su0("bar")),
  pr0("ant.cbor")]
])

113([[],
 ["http://packed.example/", /0/
  ".html"],
 [pr0(su1("foo")),
  pr0(su1("bar")),
  pr0("ant.cbor")]
])
Elements

- Table setup tag (unchanged, "overflow" model)
- Sharing reference (unchanged)
- Argument reference
  - 2 variants: fn0 ("prefix"), fn1 ("suffix")
  - works on pairs of string, array, map as before
  - new meaning consuming a tag ("function tag")
Processing model

- Packing
  - Not really a defined processing model

- Unpacking
  - Accessor-in-place (AIP) model ("pointer chasing")
  - Unpacker model
    - can be defined in terms of AIP model
Processing model: AIP(node)

— If looking at reference tag:
  — if is share-ref[n]: return aip(shared[n])
  — if is arg-ref V[n](content):  # (pr: V=0, su: V=1)
    ref = aip(argument[n])
    rump = aip(content)
    return argument_process(V, ref, rump)
— Else return item
argument_process(V, ref, rump)

If String === ref && String === rump
    return stringcat(V, ref, rump)
Elseif Array === ref && Array === rump
    return arraycat(V, ref, rump)
Elseif Map === ref && Map === rump
    return mapmerge(V, ref, rump)
Else
    function_process(V, ref, rump)
function_process(V, ref, rump): build functions from tags

— both ref and rump might be tags
  — which of them is used as "function tag"?
— ➔ use V flag (variant):
  — tags in ref are more efficient (sharing!) ➔ V=0
  — tags in rump are more flexible ➔ V=1
— If V points to non-tag: invalid
mid-fix example

["https://packed.example/foo.html",
"coap://packed.example/bar.cbor",
"mailto:support@packed.example"]

113([ /V=1 in suN/
 [],
 ["packed.example"],
 [su0(mid-fix(["https://", "/foo.html"])),
 su0(mid-fix(["coap://", "/bar.cbor"])),
 su0("mailto:support@"))]

113([ /V=0 in prN/
 [],
 [mid-fix("packed.example")],
 [pr0(["https://", "/foo.html")],
 pr0(["coap://", "/bar.cbor"]),
 pr0(["mailto:support@", "]")]}
New Terms

argument table, argument reference (was: affix)
• contrast to shared table, shared reference

function tag: defines argument reference processing
Function tags as extension point

Generic unpacker vs. application-specific function tag

— Generic unpacker knows about unknown function tag
  — due to arg-ref semantics:
    If V points to tag, MUST be a function tag
— present it to the application how?

A new tag might imply arg-ref semantics as well?
→ Harder to do forward compatibility
Packed -- separate work?

Sequences (RFC8742)

unwrap-splicing

Further function tags

mid-fix? CURIE?
(Should define at least one function tag in base doc)
Incompatible types

argument reference defined for:

— string + string (rump determines text vs. byte)
— array + array
— map + map
— V=0: tag + any, V=1: any + tag

✔: Error ➔ invalid
Define any of these behaviors (e.g., widen to array?)
Tag 6 hack

Tag 6 is efficient (1+0 byte)
Tag 6 + int ➔ shared reference (16+: 48 × 1+1, ...)
Tag 6 + string/array/map ➔ argument reference

What if V=0 function wants integer in the rump?

➔ define 224 as argument reference only synonym for 6 (pr0),
but allowing int for argument reference (no shared overload)
Extension points

- [✓] Can define new table setup tags
  - Particularly useful for static dictionaries
- [☆] Can define new function tags
  - E.g., could define for mid-fix, circumfix, templates
- [💣] Can define new reference tags?? No!?
Interoperability

Too many options → reduced interoperability
Today's "profiles":

— (1) Sharing only
— (2) Sharing + affix argument
  — Maybe (2a) strings-only, (2b) full, (2c) function tags

Suggest these profiles?
Add new tag11x for (1)???
Errata handling?

Errata reports so far only for RFC 8610 (CDDL) (some errata were in RFC 7049, superseded by RFC 8949)

States: 1 Reported, 1 HFDU, 3 Verified

Discussion at: https://mailarchive.ietf.org/arch/msg/cbor/0fhPGMaG1-TPmKlgKnP7iA2lb9M

Probably an editorial HFDU (2.2.3 needs to point to 3.6). Need text for the notes with that.
1.0:
#7.25 is a float16.
#6.55799(foo) is a tag 55799 with foo content

One is AI (additional information),
the other the full argument.

Maybe distinguish this by presence of the parentheses?

2.0: Maybe define better syntax for tags?
(CDDL 2.0: new syntax for tags with computed tag numbers?)

tagX<X, foo> = #6.X(foo)

BRZZT.

arg-ref-no<N> = 224 .plus N
arg-ref<N, arg> = #6.arg-ref-no<N>(arg)

^^^ = Syntactical pain.

^^^ = Syntactical pain.
CDDL for weird formats?

CDDL's base data model as used in the prelude (#n.nn) is good for
— CBOR,
— and, by transform from/to CBOR, JSON.

What about data models of weirder structured formats?
— RFC 8941 HTTP structured field values (SF)
— RFC 9116 security.txt?
— <insert other nightmare here>?