EAT Media Types

draft-ietf-rats-eat-media-type-01

RATS WG, IETF 115, London
Quick recap
## Base Types

<table>
<thead>
<tr>
<th>Name</th>
<th>Template</th>
</tr>
</thead>
<tbody>
<tr>
<td>EAT CWT</td>
<td>application/eat-cwt</td>
</tr>
<tr>
<td>EAT JWT</td>
<td>application/eat-jwt</td>
</tr>
<tr>
<td>Detached EAT Bundle CBOR</td>
<td>application/eat-bun+cbor</td>
</tr>
<tr>
<td>Detached EAT Bundle JSON</td>
<td>application/eat-bun+json</td>
</tr>
<tr>
<td>EAT UCCS</td>
<td>application/eat-ucs+cbor</td>
</tr>
<tr>
<td>EAT UJCS</td>
<td>application/eat-ucs+json</td>
</tr>
</tbody>
</table>
Changes since IETF 114

- draft-lundblade-eat-media-type adopted
- [Issue#4] Renamed profile to eat_profile for consistency with EAT
- [Issue#8] The DEB acronym is gone from EAT: shorthand is now "bun" from bundle
- [Issue#7], [Issue#9] Incorporate editorial suggestions from Carl and Dave
Open Issues

→ [Issue#6] - new CoAP option for carrying the profile information (*non-blocking*)
→ [Issue#10] - use cases and rationale
→ [Issue#12] - early allocation of the media types
Early Allocation Questionnaire

✓ The code points must be from a space designated as “RFC Required”, “IETF Review”, or “Standards Action”. Additionally, requests for early assignment of code points from a “Specification Required” registry are allowed if the specification will be published as an RFC.

✓ The format, semantics, processing, and other rules related to handling the protocol entities defined by the code points (henceforth called “specifications”) must be adequately described in an Internet-Draft.

✓ The specifications of these code points must be stable; i.e., if there is a change, implementations based on the earlier and later specifications must be seamlessly interoperable.

? The Working Group chairs and Area Directors (ADs) judge that there is sufficient interest in the community for early (pre-RFC) implementation and deployment, or that failure to make an early allocation might lead to contention for the code point in the field.
Next Steps (find the bug, win a beer!)

```go
func nextSteps(earlyAllocation bool) {
    waitChan := make(chan bool)

    if earlyAllocation {
        go IANA(waitChan)
    }

    for _, issue := range openIssues {
        if issue.isBlocking {
            sendPR(issue)
        }
    }

    <-waitChan

    submit(0x02)
}

func IANA(waitChan chan bool) {
    fmt.Println("do it!")

    // actually do it
    waitChan <- true
}
```