Two JW* Issues

Background

- I did a full, detailed review of the docs
- Chairs asked that these two be discussed
- More issues coming soon

Two Issues

1. Public key formats
2. COMSEC requirements for jku/x5u

Key Formats

EC + RSA

- EC keys are points on elliptic curves
  - Two coordinates from a finite field
  - Finite field can be “binary-like” or “integer like”
- RSA keys are sets of large unsigned integers

RSA Format (current)

- It is represented as the base64url encoding of the value's unsigned big endian representation as an octet sequence. The array representation MUST NOT be shortened to omit any leading zero octets.
- n = 0x04030201 could be encoded as 0x0000000000000004030201

RSA Format (issues)

- Developer-hostile: If I try to check the length of the key without decoding it, I could think it's longer than it is.

  -
    ```
    if (jwk.n.length > 342)
    {
      /* It's at least
         a 2048-bit key */
    }
    ```

RSA Format (issues)

- Incompatible with base64 padding removal: RSA keys can be any number of bits long, not just a multiple of 8
  - So if you strip the base64 padding, you don’t know how long the key is

RSA Format (proposed)

- `It is represented as the base64url encoding of the value's unsigned big endian representation as an octet sequence.  The array representation MUST utilize the minimum number of octets to represent the value.`
- `If the length of the modulus is not a multiple of 8, then it MUST be padded to the nearest multiple of 8 with leading 0 bits.`
- (That is, make “n” the same as “e”)

EC Format (current)

- `The "x" (x coordinate) member contains the x coordinate for the elliptic curve point.  It is represented as the base64url encoding of the coordinate's big endian representation as an octet sequence.`

EC Format (issues)

- Lots of ambiguities here(e.g., what is the "big-endian representation" of a finite field element?)
- The "SEC1" format is the most common standard for EC points
  - CMS, TLS, IPsec, X.509
  - ANSI X9.62, FIPS 186-2, IEEE 1363
- Let's just use that!

EC Format (proposed)

-
  ```
  {
     “kty”: “EC”,
     “crv”: “P-256”,
     “pt”: base64([0x04] || X || Y)
  }
  ```
- Allow compressed / uncompressed?

URI COMSEC

Gedankenexperiment

- Suppose the certificate referenced by an x5u is issued by a major CA
- Do I need to use TLS for the HTTP query I use to get this certificate?
- ...?

Gedankenexperiment

- Suppose the certificate referenced by an x5u is issued by a major CA
- Do I need to use TLS for the HTTP query I use to get this certificate?
- **No! The certificate is self-protecting**

URIs (current)

- The protocol used to acquire the resource MUST provide integrity protection; an HTTP GET request to retrieve the JWK Set MUST use TLS; the identity of the server MUST be validated.

URIs (issues)

- The need for TLS in this case is highly application dependent
- Some applications do need TLS
- JW* should say "this signature is valid under $KEY, which is associated to $DATA"
- It's up to the application to decide whether $DATA is enough to authenticate $KEY

URIs (proposed)

- `The protocol used to acquire the resource SHOULD provide integrity protection; an HTTP GET request to retrieve the JWK Set SHOULD use TLS.  If TLS is used, the identity of the server MUST be validated.`

Proposal Summary

- Key Formats
  - Use "SEC1" EC point format
  - Require RSA parameters to be octet strings with no leading zeros
- URIs
  - Change TLS MUST to SHOULD