Encrypted Content Type

Status Quo

AEAD ciphertext records:

```
struct {
    ContentType type;
    ProtocolVersion version;
    uint16 length;
    opaque nonce_explicit[SecurityParameters.record_iv_length];
    aead-ciphered struct {
        opaque content[TLSPlaintext.length];
    } fragment;
} TLSCiphertext;
```

Cleartext:

- `type`
- `version`
- `length`

Proposal

Distinguish TLS records with `TLS_NULL_NULL_WITH_NULL` (cleartext records) from protected TLS records.

Leave cleartext records untouched.

For non-cleartext records, move the `ContentType` inside the `ciphered struct`.

Ciphertext += 1 byte

AD -= 1 byte

Peers parse records differently depending on CCS

Backward-compatible with TLS <= 1.2 peers

Advantages

- Hides Content Type from network observer after first CCS.

```
enum {
    change_cipher_spec(20), alert(21), handshake(22),
    application_data(23), (255)
} ContentType;
```

- protects `alert` vs. `application_data`.
- hides renegotiation, if we still have it.
- hides rekeying if done with `handshake` messages
- lays groundwork for other possibly-sensitive `ContentType`s

Objections

- debugging more difficult
- decoupled network stack and TLS stack is more difficult
  - network stack can't tell when we transition from handshake to application data
- awkward repositioning of content type bytes between cleartext and ciphertext (code complexity)
- middleboxes might freak out

Options

- introduce a "dummy" header byte to placate middleboxes
- ???