Variant 1

- No DNS, 1RTT
- Plaintext SNI
  - Client -> PHDH, Client Random, Ciphersuites, Type A Extensions
  - Server <- PHDH, [PHCCS], Ciphersuite, Cert, Signed Randoms+DHParams, <CertRequest>, Type A & B Extensions
  - Client -> [PHCCS], <ClientCert>, DHParams, <SupplementalData>, Type B Extensions, <CertificateVerify>, [CCS] HTTP
  - Server <- [CCS], HTTP
- Failure Scenario:
  - Server does not support your PHDH: (2-RTT) Server jumps to Server message of Variant 3

Variant 2

- DANISH w/ B-Record, 1RTT
    - Client -> PHDH, KeyID, [PHCSS], Client Random, Ciphersuites, Type A Extensions
    - Server <- [PHCCS], Ciphersuite, Cert, Signed Randoms+DHParams, <CertRequest>, Type A & B Extensions
    - Client -> <ClientCert>, DHParams, <SupplementalData>, Type B Extensions, <CertificateVerify>, [CCS] HTTP
    - Server <- [CCS], HTTP
- Failure Scenario:
    - Server does not recognize KeyID: (2-RTT) Server jumps to Server message of Variant 3

Variant 3

- In-Bound eSNI, 2-RTT
    - Client -> Huh?
    - Server <- PHDH, KeyID
        - (This key is generic, KeyID so you can use it for later)
    - Client -> PHDH, KeyID, [PHCSS], Client Random, Ciphersuites, Type A Extensions
    - Server <- [PHCCS], Ciphersuite, Cert, Signed Randoms+DHParams, <CertRequest>, Type A & B Extensions
    - Client -> <ClientCert>, DHParams, <SupplementalData>, Type B Extensions, <CertificateVerify>, [CCS] HTTP
    - Server <- [CCS], HTTP
- Limitations
    - NIST vs Non-NIST Problem: This is not a problem if Danish is available, but we're in this situation so we assume it's not.
    - Solved by subsetting IP addresses for defaults

Remove Variant 3?

- Can't rid of the V3/2-RTT scenario, because the failure modes of V1 and V2 require it.
    - Unless the failure modes of V1 and V2 use an entirely new TLS connection, which means TCP roundtrip, which we're unwilling to do
- If we get rid of 'Huh' we have 2 (3) choices
    - Tell implementors what to put as fake data in V1 (doesn't belong in spec)
    - Tell implementors 'be creative' (hah)
    - Abandon the idea of eSNI w/o DNS data, but we'll do (b) anyway
        - The Huh? message makes is simpler for implementors to do

Load Balancers

```
Client            Load Balancer               Server


----------------------------|
                            |-------------------------|
                       decrypt, strip, pass on

This is bad.



---------------------------x---------------------------|
                       decrypt, do not modify

                       or

                       KeyID become SNI-equivalent(get back bitstring matching)

This is good.  The SNI-Equivalent is a Security Consideration.
```

Suggestions

- Servers MUST accept Variant 1, 2, or 3
- Clients SHOULD make Danish Request
    - If they receive a response, they MUST use Variant 2
    - If they do not, they MAY choose between Varient 1 & 3

Advantages of eSNI

- Advantages of doing Variant 2 (B-Records) vs Not:
  - Type A Extensions are protected
    - SNI, SRP, and others
- w/ DNSSEC protects Type A & Type B extensions against Active MITM
- Number of Client PHDH's goes from N to 1

Extensions

- Type A: Client offers, server accepts
  - Not Protected against Active or Passive MITM in Variant 1
  - Protected against Passive MITM in 2 & 3, Active MITM w/ DNSSEC
- Type B: Server offers, client accepts
  - Protected against Passive MITM in 1, 2 & 3, Active MITM w/ DNSSEC

Classifying Extensions

- Type A
  - SNI
  - signature_algorithms
  - trusted ca indication
  - server_authz
  - openpgp
  - ECC Extensions
  - SRP (Username in the clear! Security Considerations: Don't use except in Variants 2&3)
  - signature_algorithms
  - padding

- Type B
  - client certificate urls
  - truncated hmac
  - OCSP Stapling & Multi OCSP
  - user mapping
  - client_authz
  - use_srtp
  - heartbleed
  - Cert Transparency

No Type B?

- If we try to get rid of Type B extensions, all extension/negotiation offers will be in cleartext.
- If unacknowledged SupplementalData (from the Client) makes sense, that can be protected though.