CCNx Key Exchange

v5

Motivation and Goals

- We need a way to establish session keys between consumers and producers that **makes use of CCN properties**
  - Follow (D)TLS and QUIC as closely as possible
- Session keys must be **forward secure**
  - Compromising long-term secrets does not put session keys at risk
- Server-side DOS prevention (think SYN flooding)
- At most 2 RTTs to establish a session key, with the possibility for session resumption in 0 RTT
- Possible extensions for client authentication in addition to server authentication

CCNx Key Exchange Assumptions

- Consumers know the prefix of the target producer, e.g., /prefix/
- … that’s it!

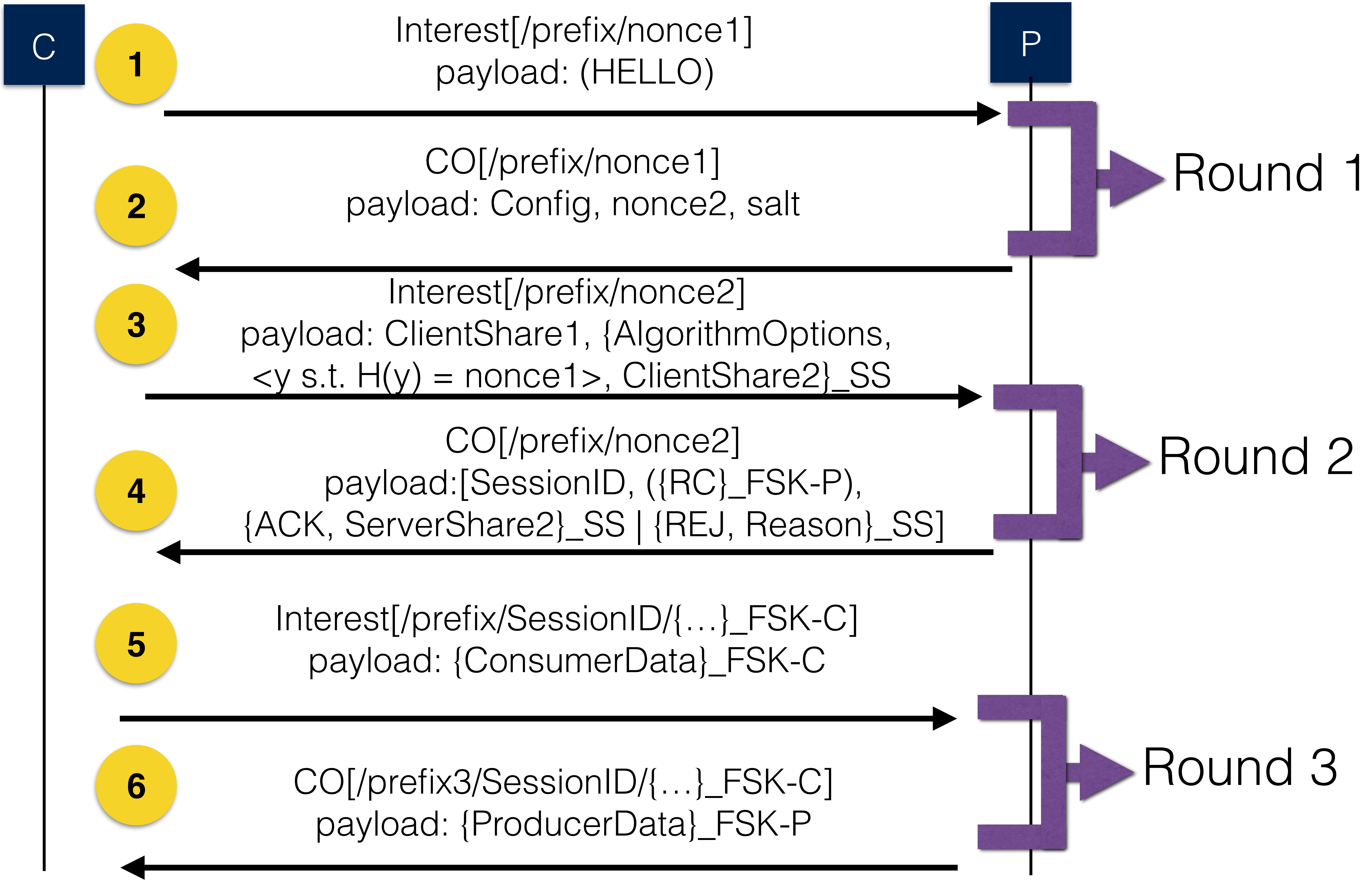
Protocol Overview

- Round 1: Obtain the server config (if not available or it has expired)

- Round 2: HELLO handshake and establish ephemeral keys

- Round 3: Final exchange to derive forward-secure secrets for all subsequent communication

Sketch of the Full Protocol



1 Interest[/prefix/nonce1]
payload: (HELLO)

2 CO[/prefix/nonce1]
payload: Config, nonce2, salt

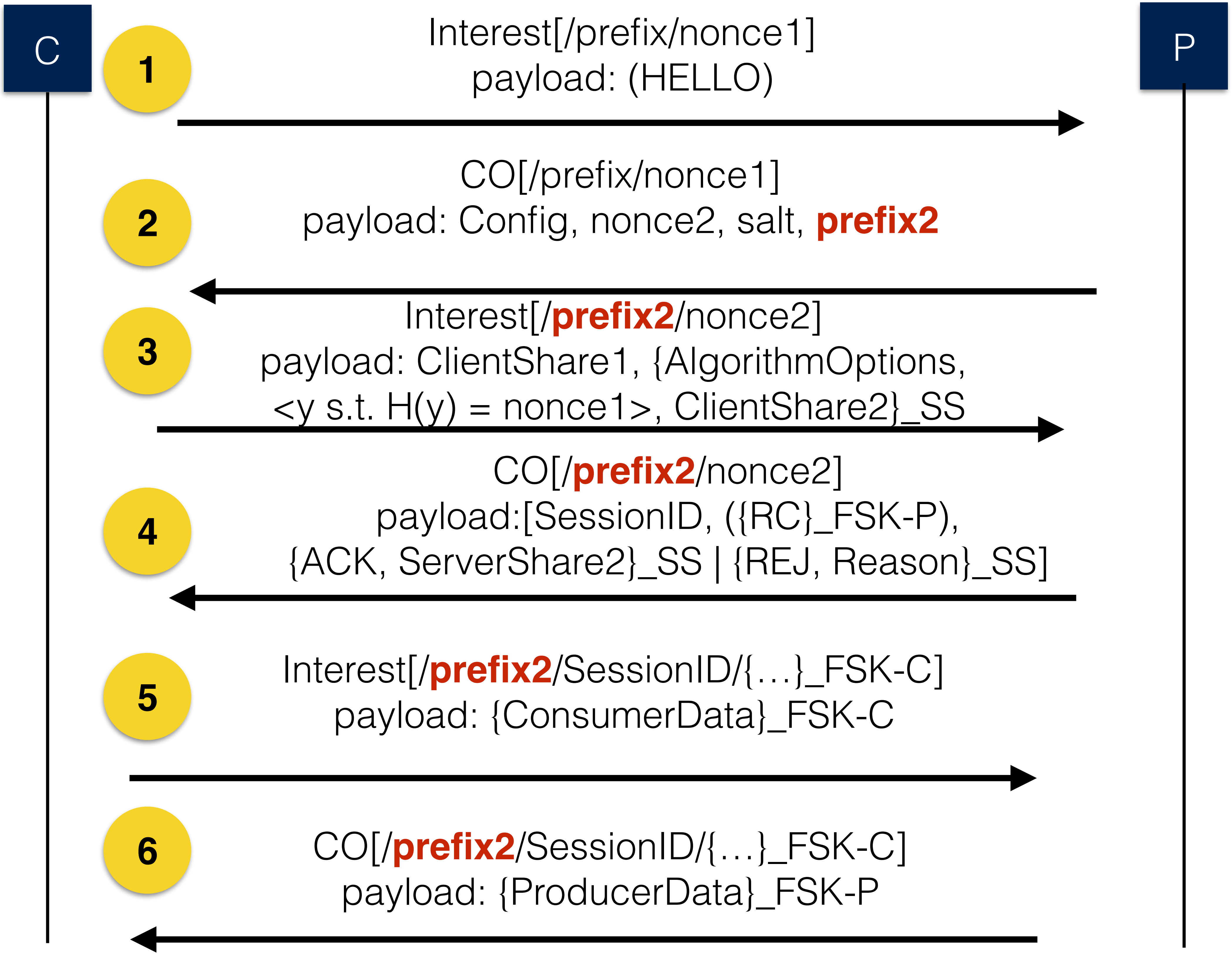
3 Interest[/prefix/nonce2]
payload: ClientShare1, {AlgorithmOptions,
<y s.t. H(y) = nonce1>, ClientShare2}_SS

4 CO[/prefix/nonce2]
payload:[SessionID, ({RC}_FSK-P),
{ACK, ServerShare2}_SS | {REJ, Reason}_SS]

5 Interest[/prefix/SessionID/{…}_FSK-C]
payload: {ConsumerData}_FSK-C

6 CO[/prefix3/SessionID/{…}_FSK-C]
payload: {ProducerData}_FSK-P

Option #1: HELLO prefix redirection



C

P

1
Interest[/prefix/nonce1]
payload: (HELLO)

2
CO[/prefix/nonce1]
payload: Config, nonce2, salt, **prefix2**

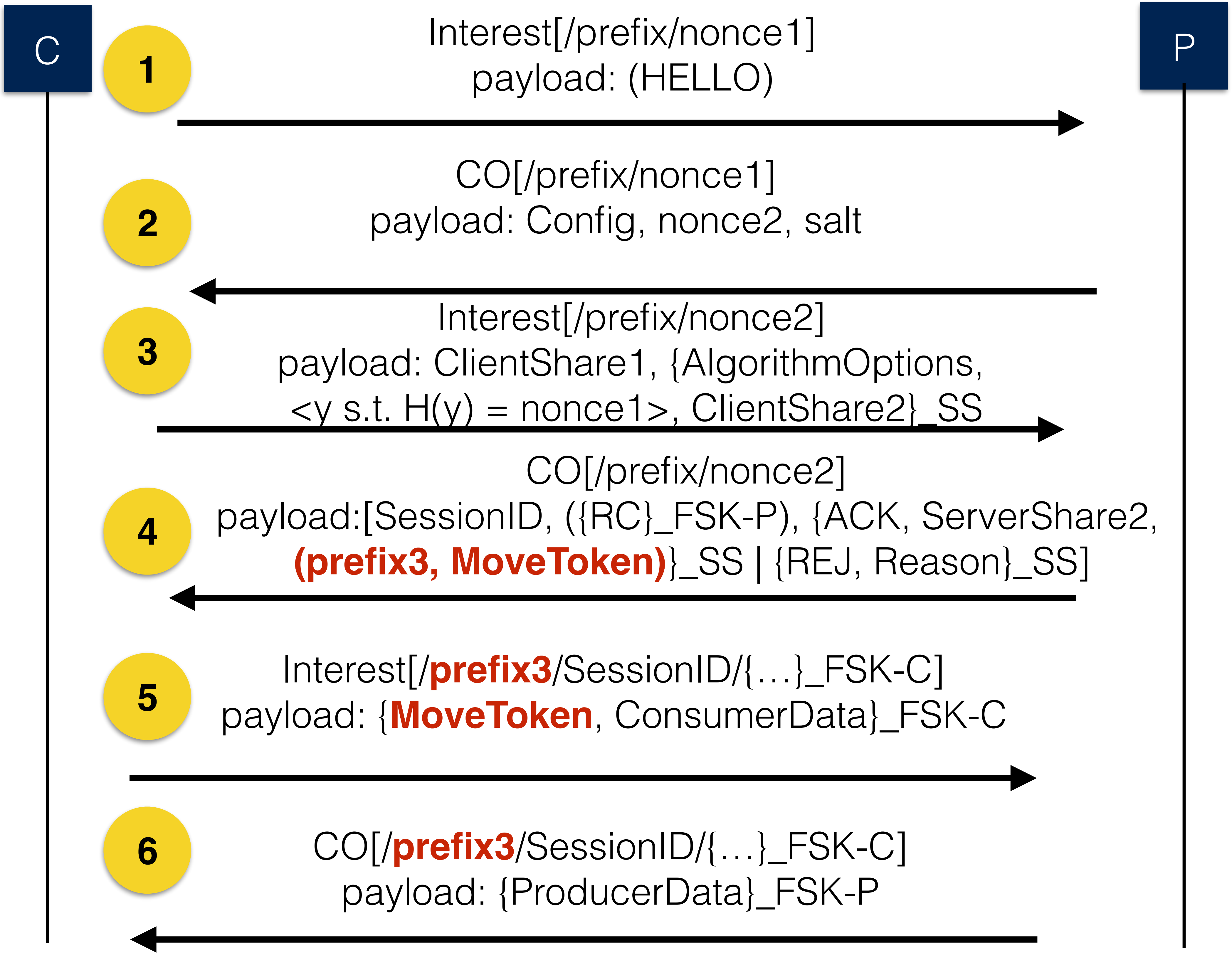
3
Interest[/**prefix2**/nonce2]
payload: ClientShare1, {AlgorithmOptions,
<y s.t. H(y) = nonce1>, ClientShare2}_SS

4
CO[/**prefix2**/nonce2]
payload:[SessionID, ({RC}_FSK-P),
{ACK, ServerShare2}_SS | {REJ, Reason}_SS]

5
Interest[/**prefix2**/SessionID/{…}_FSK-C]
payload: {ConsumerData}_FSK-C

6
CO[/**prefix2**/SessionID/{…}_FSK-C]
payload: {ProducerData}_FSK-P

Option #2: Final prefix redirection



1 Interest[/prefix/nonce1]
payload: (HELLO)

2 CO[/prefix/nonce1]
payload: Config, nonce2, salt

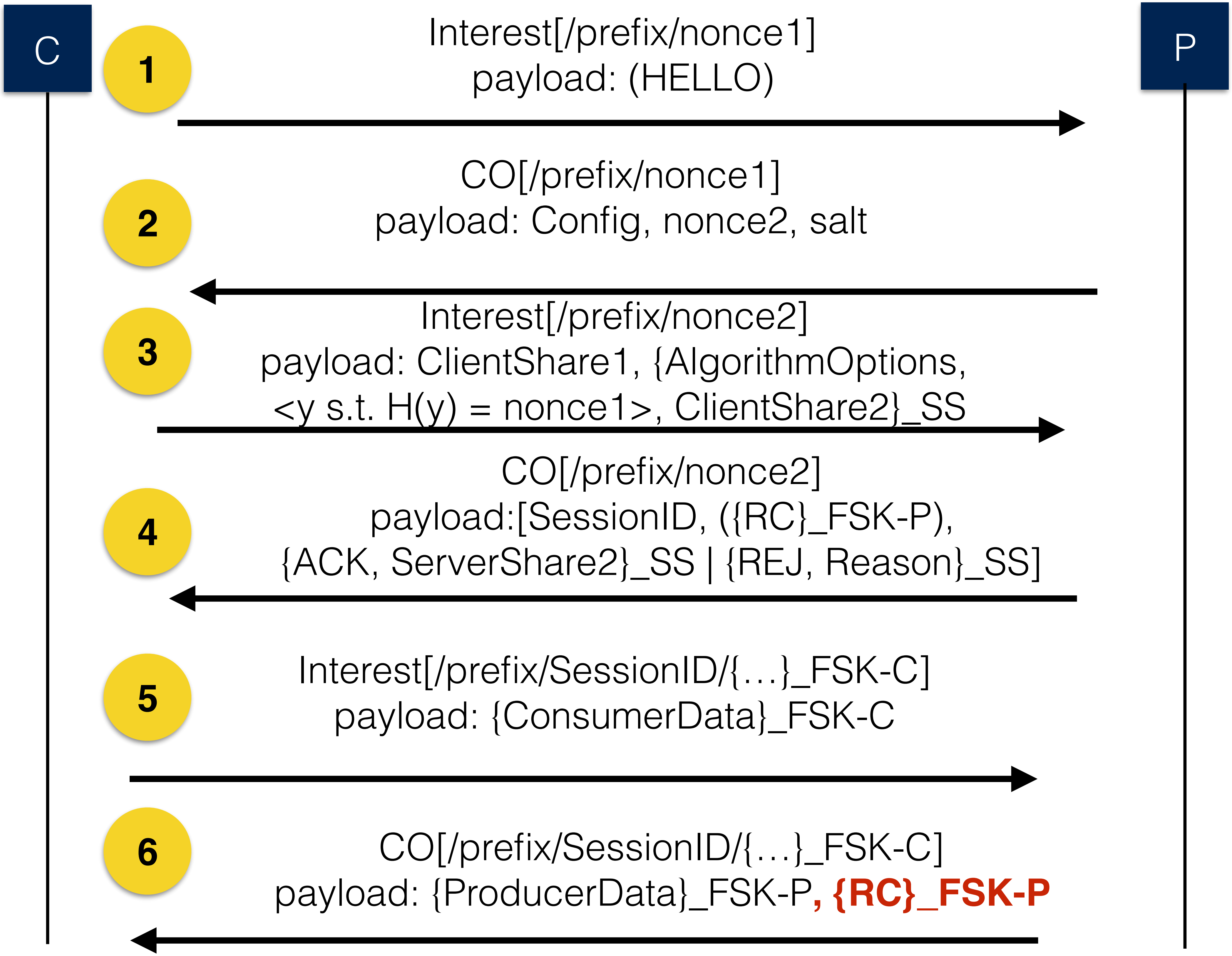
3 Interest[/prefix/nonce2]
payload: ClientShare1, {AlgorithmOptions,
<y s.t. H(y) = nonce1>, ClientShare2}_SS

4 CO[/prefix/nonce2]
payload:[SessionID, ({RC}_FSK-P), {ACK, ServerShare2,
(prefix3, MoveToken)}_SS | {REJ, Reason}_SS]

5 Interest[/**prefix3**/SessionID/{…}_FSK-C]
payload: {**MoveToken**, ConsumerData}_FSK-C

6 CO[/**prefix3**/SessionID/{…}_FSK-C]
payload: {ProducerData}_FSK-P

Option #3: Resumption Cookie Echo



1 Interest[/prefix/nonce1]
payload: (HELLO)

2 CO[/prefix/nonce1]
payload: Config, nonce2, salt

3 Interest[/prefix/nonce2]
payload: ClientShare1, {AlgorithmOptions,
<y s.t. H(y) = nonce1>, ClientShare2}_SS

4 CO[/prefix/nonce2]
payload:[SessionID, ({RC}_FSK-P),
{ACK, ServerShare2}_SS | {REJ, Reason}_SS]

5 Interest[/prefix/SessionID/{…}_FSK-C]
payload: {ConsumerData}_FSK-C

6 CO[/prefix/SessionID/{…}_FSK-C]
payload: {ProducerData}_FSK-P, **{RC}_FSK-P**

Client Authentication

- Approach 1: Provide certificate and signature in Full HELLO message

- Approach 2: Challenge-response (challenge provided in the FULL HELLO response)

- Approach 3: Plug in existing approaches (e.g., EAP)

New Material

- Consumer-provided prefix (and session ID) in Round 2 interest

- Client authentication happens after server authentication

    - Server challenge (contained in the Round 2 Content Object) must be fresh

- Re-key solution

Session Rekeying

- Consumer or producer provides KeyUpdate message in an interest or content after Round 3 is finished

- Upon receipt of a re-key message, the traffic secret is incremented by 1 and the keys are re-derived according to section 7.3 of TLS 1.3.

```
traffic_secret_N+1 = HKDF-Expand-Label(traffic_secret_N, "traffic
   secret", "", L)

 key = HKDF-Expand-Label(Secret, phase + ", " + purpose,
                                 handshake_context,
                                 key_length)
```

Open Issues

- Identifying the minimal producer routable prefix
- …