Two drafts have been adopted as active privacypass drafts:

- draft-wood-privacypass-auth-scheme-extensions
  - Adds a new extensions parameter of PrivateToken:
    Authorization: PrivateToken token="abc..." extensions="def..."

- draft-hendrickson-privacypass-public-metadata
  - defines privately and publicly verifiable public metadata mechanisms

Common review feedback:
- More detailed privacy discussion
- Add reference code
- Add test vectors
Extensions allow subdivision of entire user population.

Deployments need to take care to measuring this.

Extensions

example ripped from Chrome IP Protection project (code link)

struct **ExpirationTimestamp** {
    uint64_t timestamp_precision;
    uint64_t timestamp;
};

struct **GeoHint** {
    std::string geo_hint;
};

struct **ServiceType** {
    typedef uint8_t ServiceTypeId;
    ServiceTypeId kChromeIpBlinding = 0x01;
    ServiceTypeId service_type_id;
};

struct **DebugMode** {
    typedef uint8_t Mode;
    static constexpr Mode kProd = 0x00;
    static constexpr Mode kDebug = 0x01;
    Mode mode;
};

struct **ProxyLayer** {
    typedef uint8_t Layer;
    static constexpr Layer kProxyA = 0x00;
    static constexpr Layer kProxyB = 0x01;
    Layer layer;
};

~800 North American values

Fixed per deployment

Fixed per deployment

Fixed per verifier

Rolling groups - next slide!
ExpirationTimestamp

- Expiration and key rotation are interchangeable
- Key rotation is difficult to perform quickly
- For nonceless deployments, fast expiration is essential
- [draft spec]

```c
struct ExpirationTimestamp {
    uint64_t timestamp_precision;
    uint64_t timestamp;
}

all readers should verify safe rounding
```
Next

- Iterate on the two active drafts
- Consider an expiration extension adoption
- Re-run the CFRG adoption call of draft-amjad-cfrg-partially-blind-rsa