

Privacy Pass Extensions

Scott Hendrickson with help from Christopher Wood, Ghous Amjad, Kevin Yeo, Steven Valdez

IETF 119 - Privacy Pass

Background

- Two drafts have been adopted as active privacy pass drafts:
- [draft-wood-privacy-pass-auth-scheme-extensions](#)
 - Adds a new extensions parameter of PrivateToken:
Authorization: PrivateToken token="abc..." extensions="def..."
- [draft-hendrickson-privacy-pass-public-metadata](#)
 - defines privately and publicly verifiable public metadata mechanisms
- Common review feedback:
 - More detailed privacy discussion
 - Add reference code
 - Add test vectors

Extensions

example ripped from
Chrome IP Protection
project ([code link](#))

```
struct ExpirationTimestamp { Rolling groups - next slide!  
    uint64_t timestamp_precision;  
    uint64_t timestamp;  
}  
struct GeoHint { ~800 North American values  
    std::string geo_hint;  
};  
struct ServiceType { Fixed per deployment  
    typedef uint8_t ServiceTypeId;  
    ServiceTypeId kChromelpBlinding = 0x01;  
    ServiceTypeId service_type_id;  
};  
struct DebugMode { Fixed per deployment  
    typedef uint8_t Mode;  
    static constexpr Mode kProd = 0x00;  
    static constexpr Mode kDebug = 0x01;  
    Mode mode;  
};  
struct ProxyLayer { Fixed per verifier  
    typedef uint8_t Layer;  
    static constexpr Layer kProxyA = 0x00;  
    static constexpr Layer kProxyB = 0x01;  
    Layer layer;  
};
```

Extensions allow
subdivision of
entire user
population.

Deployments need
to take care to
measuring this.

ExpirationTimestamp

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

```
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](#)