# Blind Signatures Scheme for PCM Fraud Prevention

## W3C PrivacyCG F2F - May 2021


User

The Click Source

The Click Destination

Client visits website

Click source returns ad links with attribution attributes

<a attributionsourceid=3 attributiondestination="https://shoes.example" attributionsourcenonce="ABCDEFabcdef0123456789">Buy great shoes now! </a>

Clicks on link

GET https://clicksource.example/.well-known/private-click-measurement/get-token-public-keys/

Public Key of the Click Source

clientNonce = random(32)
(blinded_msg, inv) =
Blind(pkCS, clientNonce)

POST https://clicksource.example/.well-known/private-click-measurement/sign-unlinkable-token/
{ "key_id": …, "source_engagement_type": "click", "source_nonce": ABCDEFabcdef0123456789,
"source_unlinkable_token": blinded_msg , "version": 2}

Validates the source nonce is valid
And computes
blind_signature =
BlindSign(skCS, blind_msg)

"unlinkable_token": blind_signature

TokenSignature =
Finalize(pkCS, clientNonce, blind_signature, inv)
Store TokenSignature

Triggering event happens and
$r \rightarrow [24,48$ hours] passes

GET https://clicksource.example/.well-known/private-click-measurement/get-token-public-keys/

Public Key of the Click Source

Client verifies pkCS is still
a key of the click source

Attribution report is sent to clickSource and clickDestination, containing the clickSource, clientNonce …

GET https://clicksource.example/.well-known/private-click-measurement/get-token-public-keys/

Public Key of the Click Source

Verifies uniqueness of clientNonce and
validity of the signature of the
attribution report

Verifies uniqueness of clientNonce and
validity of the signature of the
attribution report

Issue #41 on GitHub.com/privacycg/private-click-measurement

# Cryptographic Scheme for Fraud Prevention

# Requirements for the cryptographic scheme

- **Unlinkability** between the issuance of the token with the source nonce, and the redemption flow when the attribution report is posted
- **Public verifiability** to allow both the click source and destination to verify the authenticity of the report
- **Resistance against “one-more-forgery” attacks**, to prevent an attacker from minting valid signatures for attribution reports

# Additional Considerations

- **Efficiency:**
  - *Computational*: The amount of computation on device and on the server is reasonable.
  - *Bandwidth*: The amount of data transmitted is reasonable.
- **Ease of Implementation**
- **Post-Quantum Security**

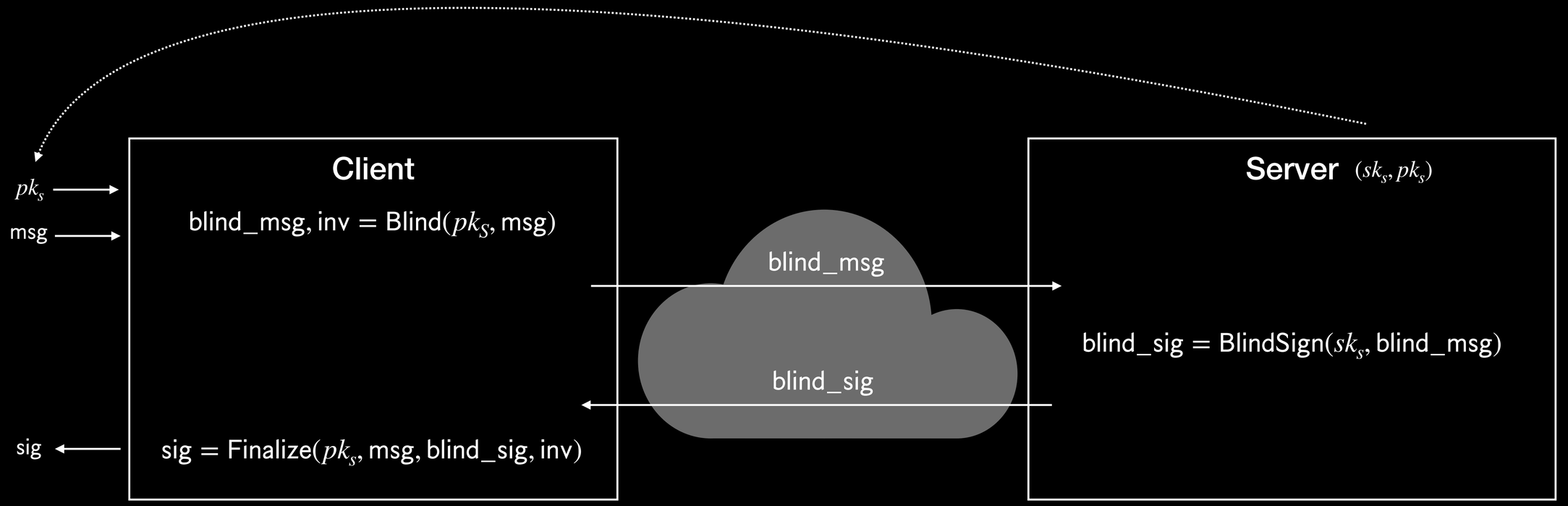
💡 **Cryptographic scheme is going to be versioned to allow future clients to use updated cryptographic primitives.**

# RSABSSA

## IETF Draft

- Based on Chaum's Blind Signatures (been around since 80s and well-studied)
- Publicly-verifiable RSA-PSS signatures
  - Widespread support in crypto libraries to verify those signatures
  - Resistance against "one-more-forgeries"
- Unlinkability against classical and quantum attackers.
  - Forgery believed to be possible by future quantum attacker.
- Implementation is straightforward for crypto libraries that have already RSA support

# Blind RSA
## Protocol



# Alternatives Considered

| Scheme | Pros | Cons |
|---|---|---|
| ECVOPRFs | Smaller key sizes, more efficient | Not publicly verifiable |
| Blind Schnorr Signatures | Smaller key sizes, more efficient<br>Threshold-friendly (c.f. FROST) | Three messages (state or computation overhead)<br><br>Polynomial-time ROS attack (2020/945), but FPS20 (Clause-blind Schnorr) seems plausible, but would require multiple round-trips for issuance and hasn't received much peer-review yet. |
| Blind BLS | Smaller key sizes | Expensive signing and verification<br><br>Pairing support is not (yet!) widely supported in common libraries (BoringSSL, ring, etc) |
| Abe | Polynomial concurrent security<br>Seems unaffected by ROS attack (2020/945) | Three messages (state or computation overhead)<br>Large signature sizes (several group elements) |