Rate-Limited Issuance

Tommy Pauly & Chris Wood

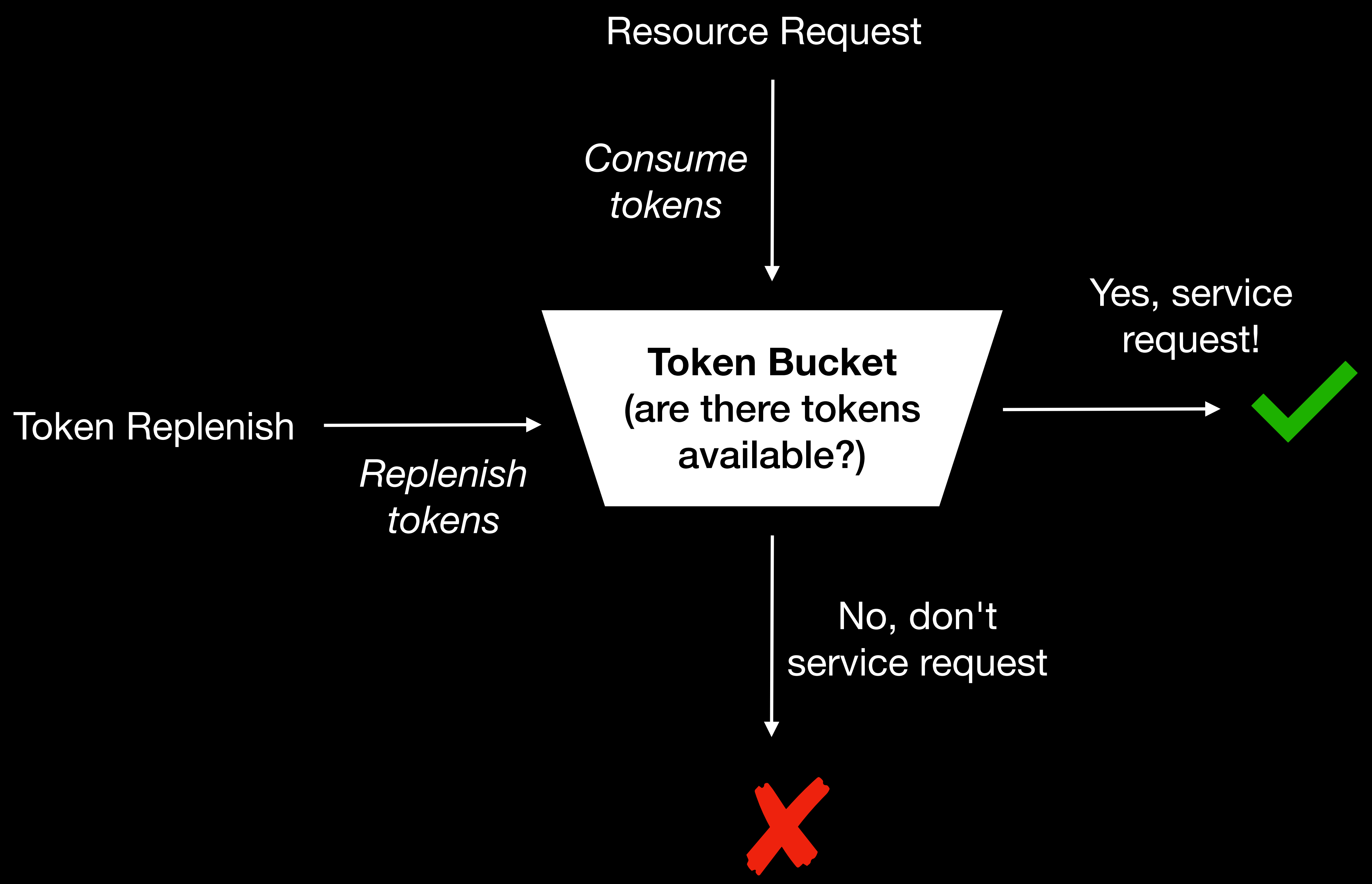
Rate-limiting is a common part of fraud prevention and anonymous access

It also often relies on tracking cookies or client IP addresses

A common way to implement this is with "token buckets"

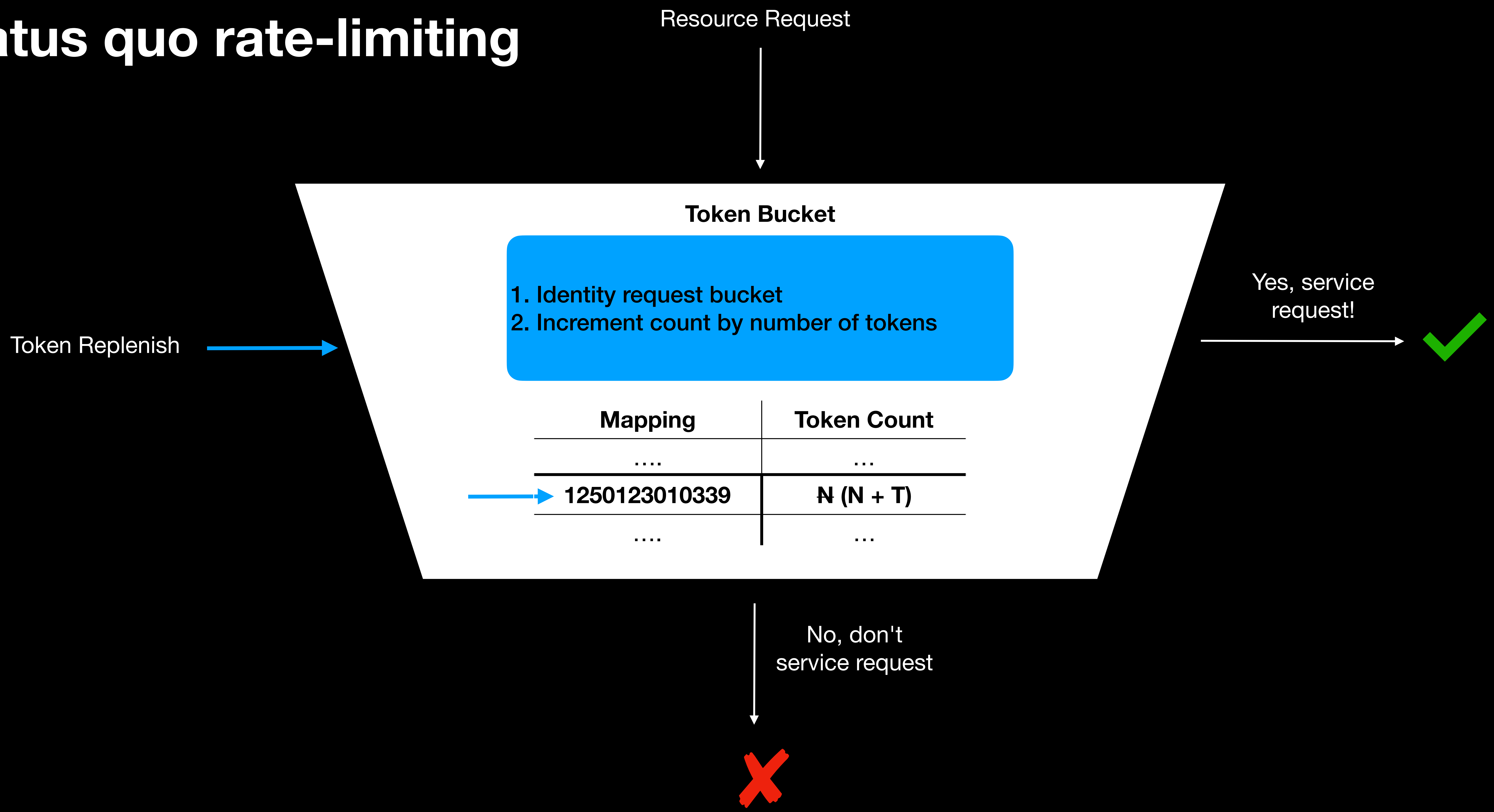
Token buckets

Status quo rate-limiting



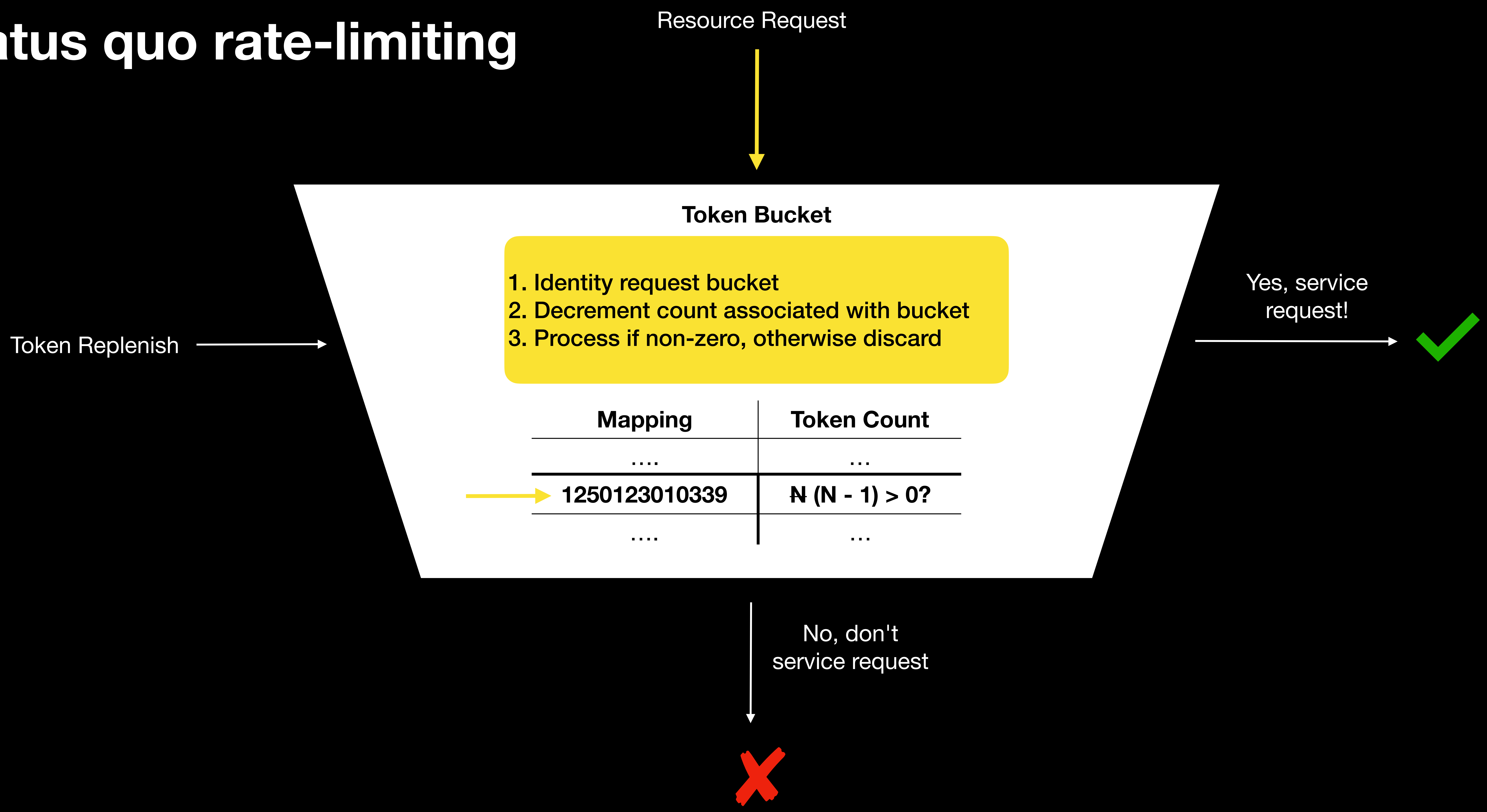
Token buckets

Status quo rate-limiting



Token buckets

Status quo rate-limiting



Why Privacy Pass?

Existing rate-limiting schemes break down when clients have more privacy (*shared rate-limiting buckets*)

- Proxies
- VPNs
- Shared IPs on public networks

A basic Privacy Pass token isn't always enough

- Attests to the fact that a device or user passed some check
- Does not prevent that legitimate device or user doing too many actions (click farm, or abuse of metered paywall)
- Degenerates to blocking access

Private Token variants

Basic Token (OPRF, RSA Blind Signature)	Rate-Limited Token (RSA Blind Signature)
Attests to user/device legitimacy	Attests to user/device legitimacy **+ access rate below threshold**
Replaces captcha for improving confidence in user	Adds mitigations against a device in a click farm
	Allows metered paywall access

Rate-Limited Tokens

Rate-limited tokens *extend* the basic issuance protocol with new properties:

1. Attester maintains counters for client + anonymized origin
    - Attesters learn ***stable mapping*** between per-client secret and per-origin secret, without learning only per-origin information
2. Issuer provides a rate limit to enforce when issuing tokens
    - Issuers learn origin associated with a token challenge, encrypted with HPKE
3. Attesters fail requests if the per-origin rate limit is exceeded

Rate-Limited Tokens

Rate-limited tokens *extend* the basic issuance protocol with new properties:

1. Attester maintains counters for client + anonymized origin
    - Attesters learn ***stable mapping*** between per-client secret and per-origin secret, without learning only per-origin information

This is the main challenge for the protocol

2. Issuer provides a rate limit
    - Issuers learn origin associated with a token challenge, encrypted with HPKE
3. Attesters fail requests if the per-origin rate limit is exceeded

Attester state

The "token buckets" used for rate limiting now are "**private** token buckets" maintained on the attester

- A stable mapping is a deterministic function between per-client and per-origin information, e.g., F(client secret, origin secret)
- The mapping is used to enforce rate limits based on individual clients for individual origins
- Attester uses mapping as index into data structure tracking per-client state

F(client secret, origin secret) →

Mapping	Count
….	**…**
12311235123	N
….	…

Stable Mappings and Rate Limits

Stable Mappings and Rate Limits

Current Status

Builds on signature schemes with key blinding for private computing the stable mapping

Requires split deployment model for meaningful privacy

Several interoperable implementations exist and security analysis is underway

Is the WG interested in adopting this draft?

Backup

Rate-Limited Tokens

Rate-limited tokens *extend* the basic issuance protocol with new properties:

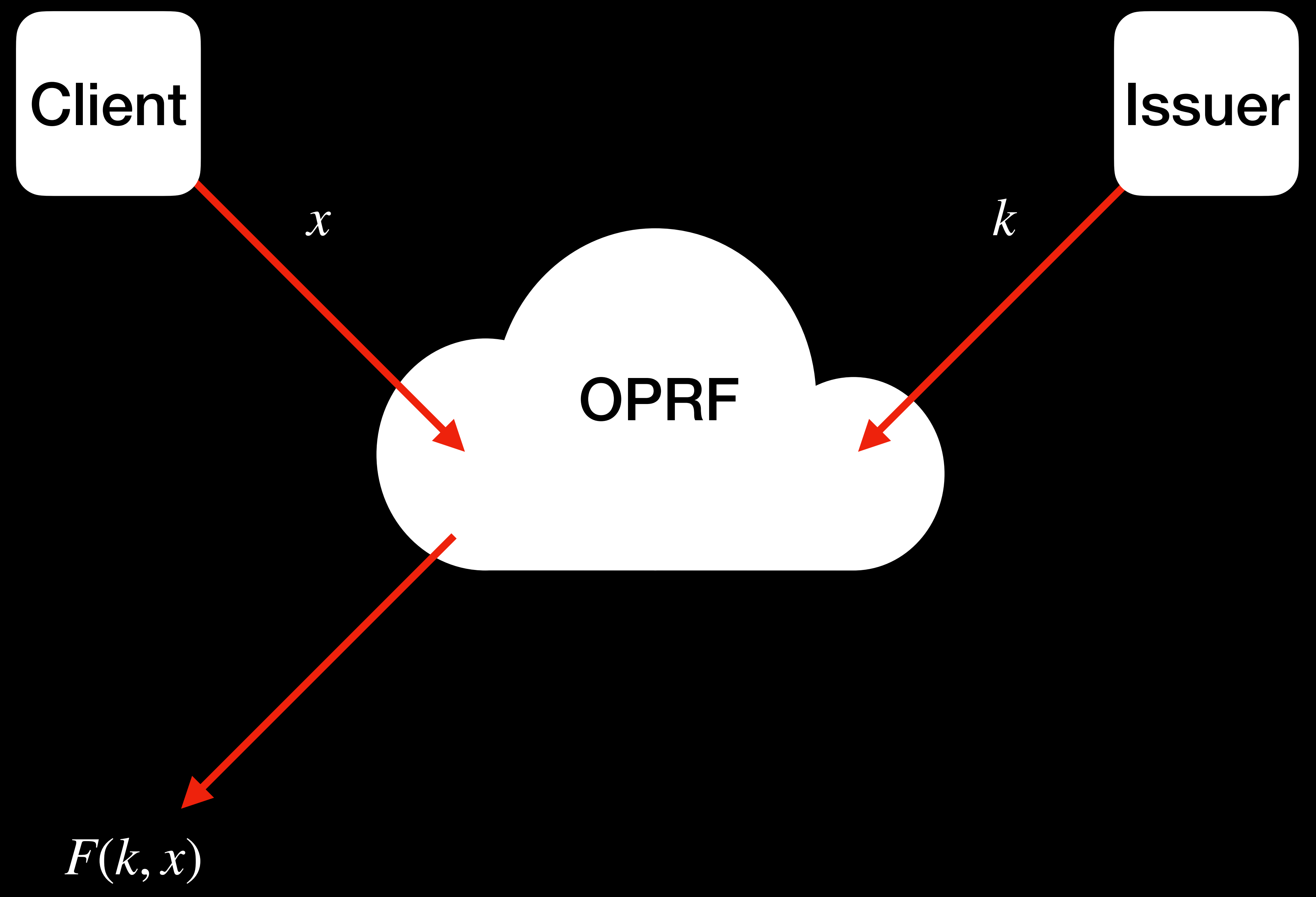
1. Attester maintains counters for client + anonymized origin
   - Attesters learn ***stable mapping*** between per-client secret and per-origin secret, without learning only per-origin information

Can we use an OPRF to compute this?...

2. Issuer provides a rate limit
   - Issuers learn origin associated with a token challenge, encrypted with HPKE
3. Attesters fail requests if the per-origin rate limit is exceeded

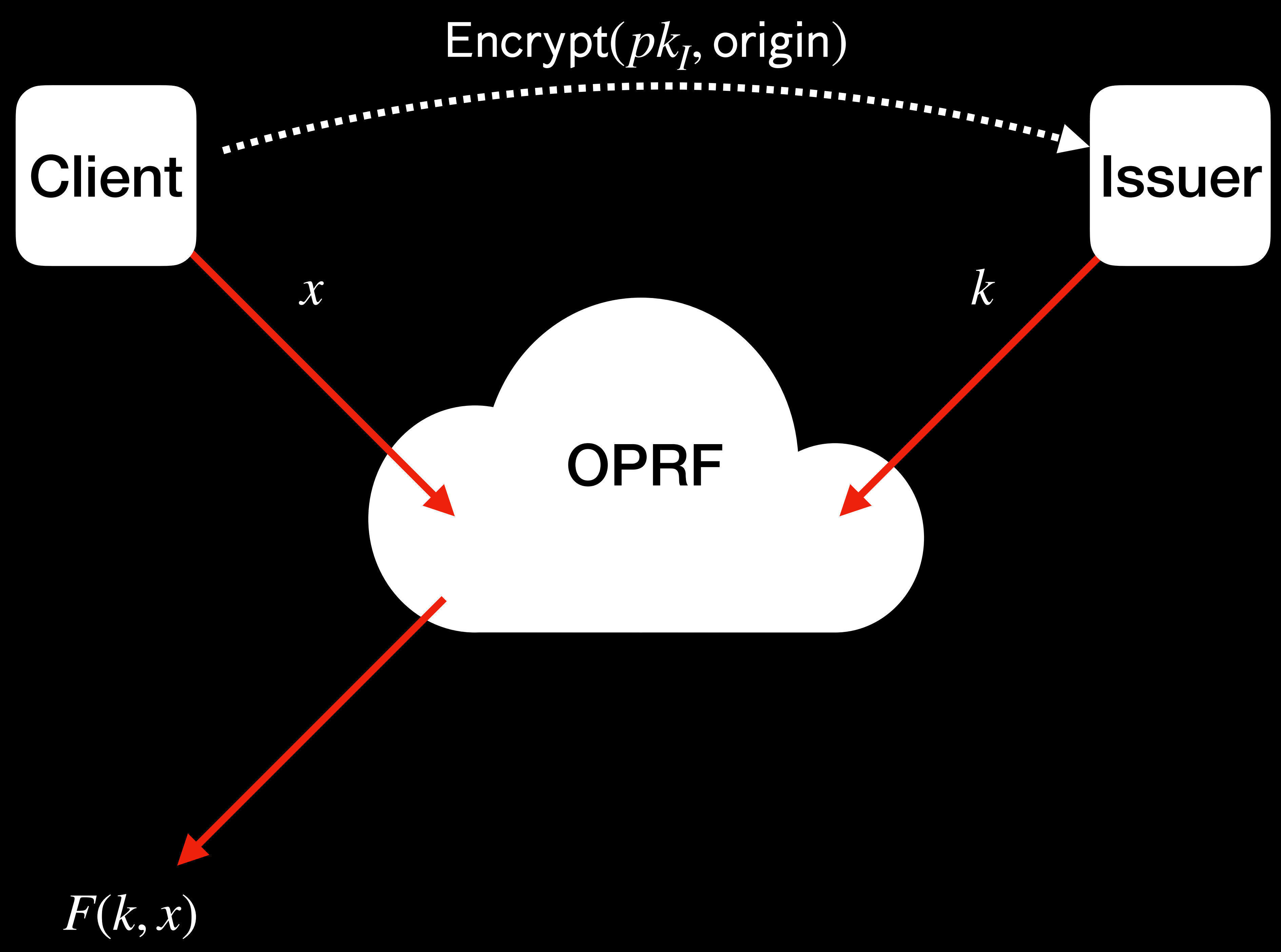
An OPRF Sketch

An OPRF protocol computes $F(k, x)$ for per-origin k and per-client x



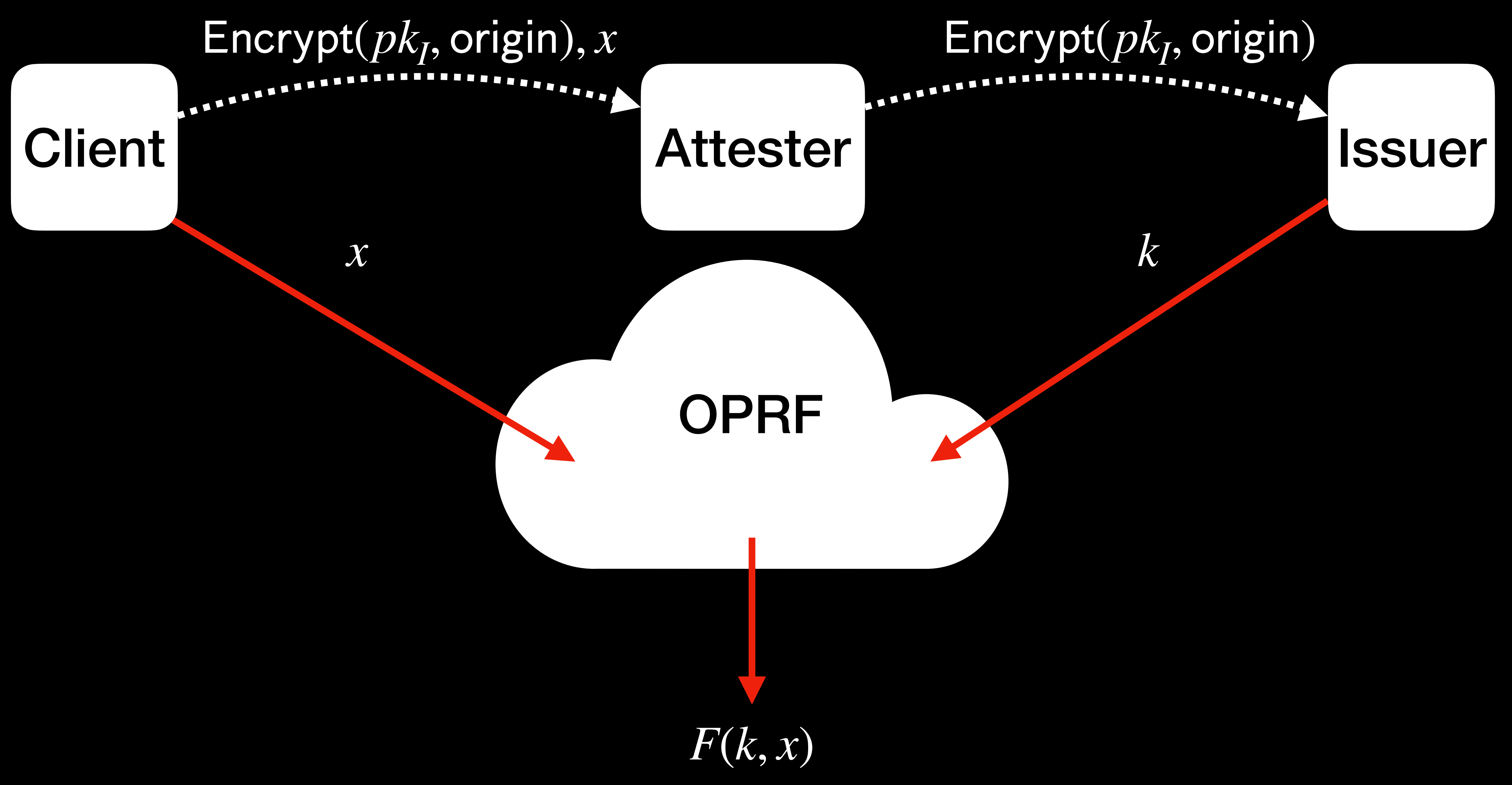
An OPRF Sketch

Clients can encrypt the origin identifier under the Issuer's public key



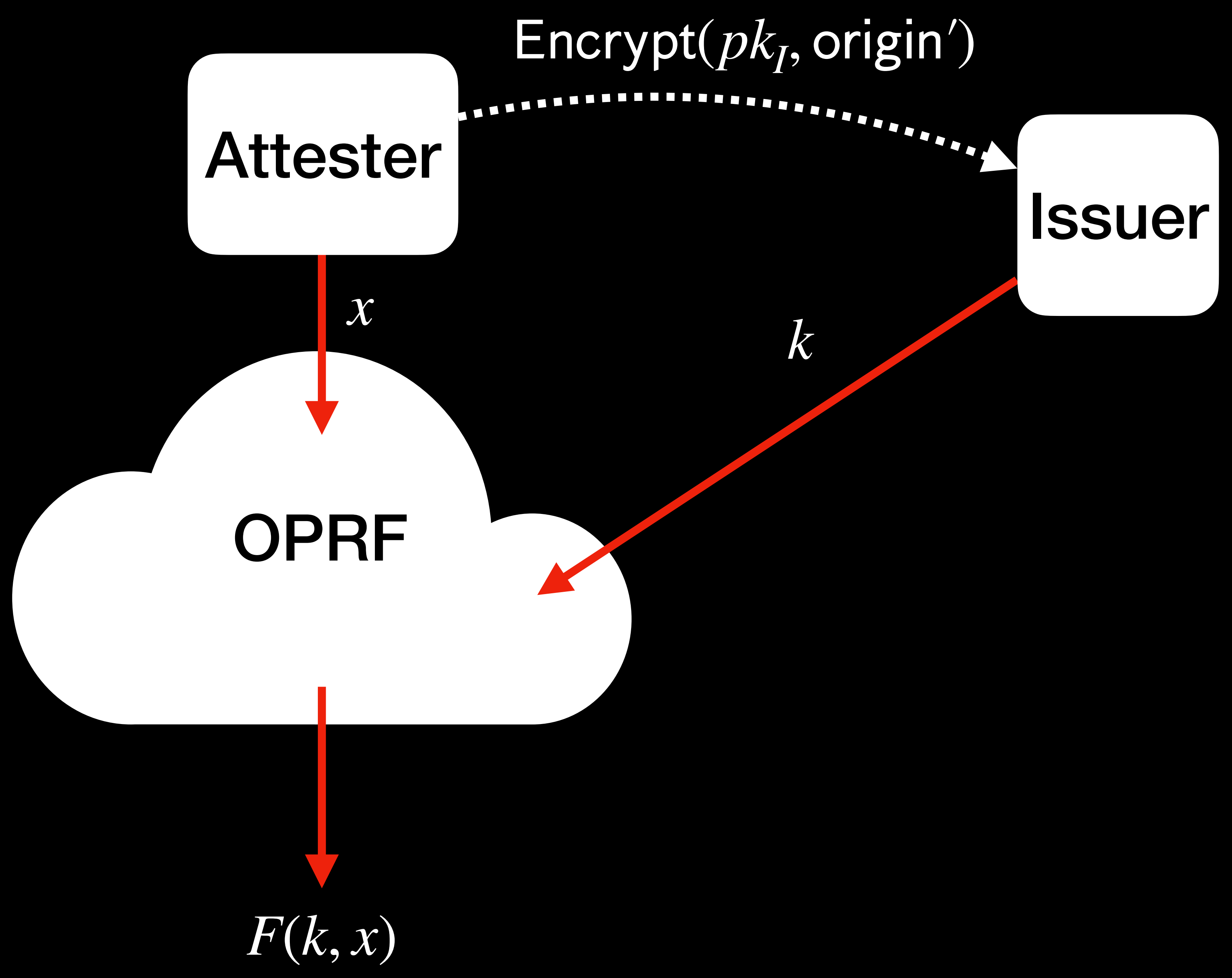
An OPRF Sketch

An Attester can relay the encrypted origin name and complete the OPRF



An OPRF Sketch

… Attester can perform a dictionary attack to learn $F(k, x)$



Rate-Limited Tokens

Rate-limited tokens *extend* the basic issuance protocol with new properties:

1. Attester maintains counters for client + anonymized origin
    - Attesters learn ***stable mapping*** between per-client secret and per-origin secret, without learning only per-origin information
2. Issuer provides a rate limit
    - Issuers learn origin assoc … HPKE
3. Attesters fail requests if the per-origin rate limit is exceeded

An OPRF alone isn't sufficient because of dictionary attacks. Computation of the mapping requires proof of ownership for the per-client secret.

Signature Scheme with Key Blinding

Extend digital signature schemes with two functionalities for *signing requests*

BlindPublicKey and UnblindPublicKey: Given public key and secret blind, produce *blinded public key*

UnblindPublicKey(BlindPublicKey(pkS, skB), skB) = pkS

BlindKeySign: Sign message with secret key and secret blind

Verify(BlindPublicKey(pkS, skB), msg, BlindKeySign(skS, skB, msg)) = true

Draft specification: `https://datatracker.ietf.org/doc/draft-dew-cfrg-signature-key-blinding/`

Detour: Signature Scheme with Key Blinding

Use signature public key blinding to compute an OPRF*

Let skC (skO) be the per-Client (per-Origin) secret with public key pkC (pkO), and let skR be a random client-generated blind per request

$$F(skC, skO) = Hash(pkC, Blind(pkC, skO))$$

$$= Hash(pkC, Unblind(Blind(Blind(pkC, skR), skO), skR))$$

Close -- but not identical -- to the OPRF construction in `draft-irtf-cfrg-voprf`

* Security analysis is underway and results will be published in before IETF 114

Rate-Limited Tokens