

OAuth WG, IETF 98 Chicago

# OAuth JPoP

~ JWT PoP Token Usage ~

<https://tools.ietf.org/html/draft-sakimura-oauth-jpop-04>

March 27, 2017

Nat Sakimura, Nomura Research Institute  
John Bradley, Ping Identity  
(Kepeng Li, Alibaba Group)

We were thinking that for the token usage, Token Binding will solve all problems

**WRONG!**

# On some circumstances, it does not.

- Cannot touch the infrastructure. Everything has to be done at the application layer.
- Need somehow to tie back to the TLS client authn.
- Etc.

## And they need it now.

E.g. UK Open Banking

# Here comes the JPoP!

Parallel document to  
RFC6750.

[\[Docs\]](#) [\[txt|pdf|xml\]](#) [\[Tracker\]](#) [\[Email\]](#) [\[Diff1\]](#) [\[Diff2\]](#) [\[Nits\]](#)

Versions: ([draft-sakimura-oauth-rjwtprof](#)) [00](#)  
[01](#) [04](#)

OAuth Working Group  
Internet-Draft  
Intended status: Standards Track  
Expires: September 28, 2017

N. Sakimura  
Nomura Research Institute  
K. Li  
Alibaba Group  
J. Bradley  
Ping Identity  
March 27, 2017

**The OAuth 2.0 Authorization Framework: JWT Pop Token Usage**  
**draft-sakimura-oauth-jpop-04**

Abstract

This specification describes how to use JWT POP (Jpop) tokens that were obtained through [\[POPKD\]](#) in HTTP requests to access OAuth 2.0 protected resources. Only the party in possession of the corresponding cryptographic key for the Jpop token can use it to get access to the associated resources unlike in the case of the bearer token described in [\[RFC6750\]](#) where any party in possession of the access token can access the resource.


## The draft defines

- 3. JWT POP Token . . . . .
- 4. Sender Constrained Token . . .
  - 4.1. DN Constrained Token . . .
  - 4.2. Client ID Constrained Token
- 5. Key Constrained Token . . . . .
- 6. Resource access method . . . . .
  - 6.1. Mutual TLS access method . .
  - 6.2. Signature method . . . . .
- 7. Authorization Error . . . . .
- 8. IANA Considerations . . . . .
  - 8.1. Jpop Authentication Scheme
  - 8.2. JWT Confirmation Methods .

# JWT PoP Token looks like:

```
{
  "iss": "https://server.example.com",
  "aud": "https://resource.example.org",
  "iat": "1360189224",
  "exp": "1361398868",
  "cnf": {...}
}
```

Replay protection. (It must only span a same administrative domain.)




All names are mandatory

# CN cnf Method (CN Constrained Token)

```
{
  "iss": "https://server.example.com",
  "sub": "joe@example.com",
  "aud": "https://resource.example.org",
  "exp": "1361398824",
  "nbf": "1360189224",
  "cnf": {
    "dn": "cn=John Doe LLC,dc=client,dc=example,dc=com"
  }
}
```

The Distinguished Name of the client certificate that the client used in the authorization request.





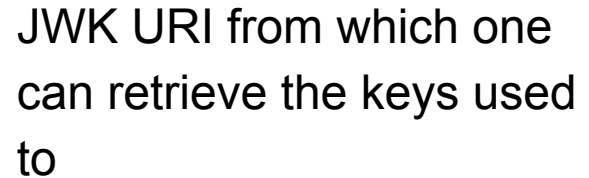
# Client ID cnf Method (cid Constrained Token)

```
{
  "iss": "https://server.example.com",
  "sub": "joe@example.com",
  "aud": "https://resource.example.org",
  "exp": "1361398824",
  "nbf": "1360189224",
  "cnf": {
    "cid": "client-id-used-in-the-token-request"
  }
}
```

# jku cnf Method (jku Constrained Token)

```
{
  "iss": "https://server.example.com",
  "sub": "joe@example.com",
  "aud": "https://resource.example.org",
  "exp": "1361398824",
  "nbf": "1360189224",
  "cnf": {
    "jku": "https://client.example.com/keys/client123-jwks",
    "Kid": "2017-03-31"
  }
}
```

JWK URI from which one  
can retrieve the keys used  
to



In addition,

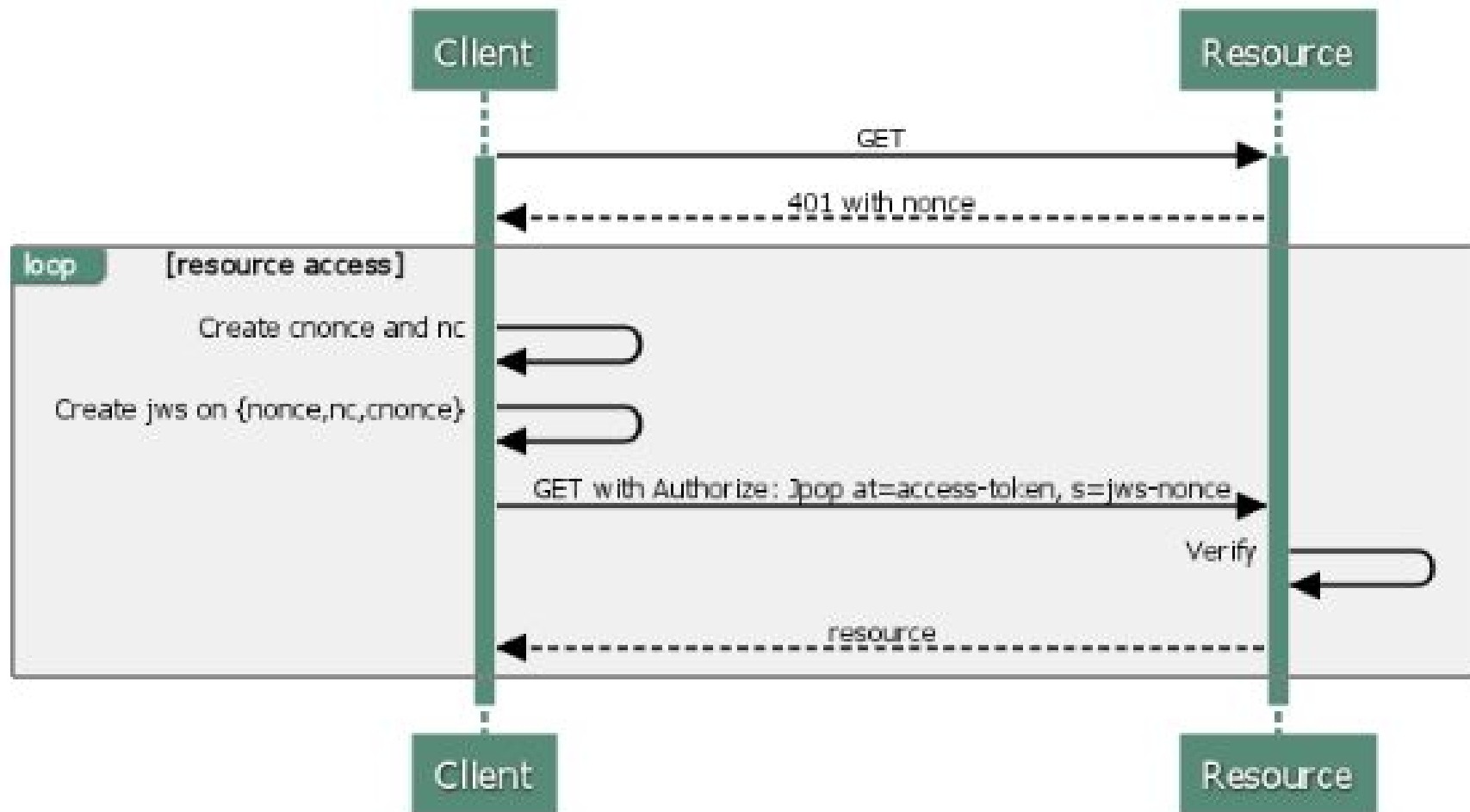
- jwk JSON Web Key Representing a Public Key
- jwe Encrypted JSON Web Key
- jwkt#s256 [RFC7638] Thumbprint of a JWK using the SHA-256 hash Function.
- x5t#s256 [RFC7515] X.509 Certificate SHA-256 Thumbprint

are defined.

# Resource Access Methods

- Mutual TLS access method
  - CN cnf Method
  - x5t#s256 cnf method
  - jku cnf method
- Signature Method
  - See the next slide

# OAuth JPOP: Signature Method



## Downside of us not standardizing it now:

- Groups like OBS are likely to start deploying a proprietary solution and will not be replaced by a standard for a long time.

Please adopt the draft as a  
WG item!