Distributed OAuth

draft-hardt-distributed-oauth

Dick Hardt
IETF 100, Singapore
November 2017
Problem

- OAuth 2 presumes **static relationship** between authorization server and protected resource that is **known a priori** by client.

- Global systems have similar protected resources, that are managed by different authorization servers. Eg. Different geopolitical regions.

- Large, distributed systems need to evolve the relationship between authorization servers and protected resources.

- Clients need to **dynamically** learn the authorization server for a given protected resource **at run time**.
Client Accessing Global Protected Resources

- Client
- .CN: S3, Authorization Server, EC2
- .US: S3, Authorization Server, EC2
- .EU: S3, Authorization Server, EC2
Proposed Solution

- Client discovers authorization server from protected resource in HTTP 401 response

HTTP/1.1 401 Unauthorized
WWW-Authenticate: Bearer
realm="example.realm",
iss="http://issuer.example.com/token",
scope="example_scope", error="invalid_token"
Threats

1. Access Token Reuse
   - Resource server uses access token at other protected resources

2. Resource Server Impersonation
   - Resource server provides meta data needed for different resource server

3. Malicious Authorization Server
   - Authorization server may replay client credentials at different authorization server
Mitigation
Access Token Reuse

- Protected Resource specific access token
  - Client provides “host” parameter in access token request that matches protected resource host in TLS certificate
  - Authorization server includes “host” parameter in access token
  - Protected Resource verifies “host” parameter in access token

- Requires an access token for each PR
Alternative Mitigation
Access Token Reuse

- Client Authentication
  - Client authenticates in call to Protected Resource
  - Protected Resource verifies client is “sub” in access token
- Requires PR to verify identity of Client
Mitigation
Resource Server Impersonation

- Same as Access Token Reuse
Mitigation
Malicious AS

- Client MUST use proof of possession mechanism to authenticate to authorization server (AS) that is resistant to man-in-the-middle attacks
- eg. Mutual TLS profile for OAuth
Next Steps

- OAuth WG interest?