Introduction

• New Dynamic OAuth Cases Emerging
  – OIDC where clients discover and connect to OPs on-the-fly
  – Email – OAuth enabled email (Kitten/SASL ext)
  – Tenancy and Published Software Providers
    • What kind of automated configuration will they use?

• How do dynamic clients get correct OAuth & resource endpoints?
  – Dyn Reg, AuthZ, Token, Resource
Why is this a Problem?

• OAuth security depends upon establishing TLS to the correct endpoints
  – When RFC6749 published, most endpoints were fixed

• A hacker that can provide a client with a set of endpoints potentially insert a MITM proxy and TLS will be valid
  – Users don't see calls to Token and Resource endpoints – no-real person sees the location is suspicious
This is not Mix-Up

• This attacks clients that configure on-the-fly to talk to new servers if discovery is insecure
• It does not require multiple OAuth configs
• Attack occurs because client is convinced to talk to the wrong TLS protected endpoints
Discovery Challenges

• Wide variety of use cases – probably impractical to define resource discovery
• A resource server MAY have more than one authorization server
  – Tenancies & Other Federated Scenarios
  – UMA user delegated authz services
• The resource server can often indicate the AS, but not always
• The AS may not know about all of the resources it protects
  – Differences between "aud" and real resource endpoints
  – Often unwilling or unable to reveal list of valid endpoints
Issues With OAuth-discovery

• Draft returns configuration (Dyn Reg, AS, Token) for OAuth endpoints only
  – OIDC includes the UserInfo endpoint!
• The same MITM attack against token endpoint can be done against resource endpoint
  • Only half the problem solved!
• Only one OAuth server can be returned per ./well-known endpoint
  – Limitations in some cloud / tenancies scenarios
Bound Config

• Introduces a "confirm" function to the discovery request
  – Does the client have a valid resource server?
• Discovery service returns config based on the requested resource
  – Can route client to a specific assigned OAuth svc
• Different AS configs can be returned based on parameter
• Can be simple with just URL pattern filter (*.rs.example.com)!
• Can be very granular
  – based on security context or full RS urls
Bound Config & Resource Indicator

• Resource Indicator draft related but complementary/different
  – Resource indicator very good for handling cross-user or shared resource scope requests
  – Bound config simply confirms the server validity

• Bound Config is one time based on course URL matching (the root server)

• Good to ensure alignment and consistency
Example Request

GET /.well-known/webfinger
?resource=https%3A%2F%2Ffinance.example.com%2Fabcpath
&rel=oauth2
HTTP/1.1
Host: example.com

- Resource parameter mainly needs the domain name or a pattern match for the server for security.
- Full path is useful if different OAuth servers are being selected based on path.
Example Response

{
    "subject": "https://finance.example.com",
    "links": [
        {
            "rel": "oauth2",
            "href": "https://server.example.com",
            "properties": {
                "issuer": "https://server.example.com",
                "authorization_endpoint": "https://server.example.com/oauth/authorize",
                "token_endpoint": "https://server.example.com/oauth/token",
                "token_endpoint_auth_methods_supported": ["client_secret_basic", "private_key_jwt"],
                "token_endpoint_auth_signing_alg_values_supported": ["RS256", "ES256"],
                "userinfo_endpoint": "https://server.example.com/oauth/userinfo",
                "jwks_uri": "https://server.example.com/jwks.json",
                "registration_endpoint": "https://server.example.com/oauth/register",
                "scopes_supported": ["openid", "profile", "email", "address", "phone", "offline_access"],
                "response_types_supported": ["code", "code token"],
                "service_documentation": "http://server.example.com/oauth/service_documentation.html",
                "ui_locales_supported": ["en-US", "en-GB", "en-CA", "fr-FR", "fr-CA"]
            }
        }
    ]
}
Error Response

• Use normal WebFinger signalling
• A failure to match resource returns nothing
• Hard Failure - The client developer can't proceed without passing a valid resource
  – Service provider can verify client's intent.
Is it Implemented?

- Web Finger is well implemented
- OIDC Discovery puts RS in Config
  - UserInfo is part of discovery
  - Technique won't work if many 1000s of RSs
- Proposal is to replace oauth-discovery draft
  - hold as part of an overall set of specifications targeted towards dynamic OAuth scenarios
    - Resource Indicator, Mix-up, ....
  - Get implementation started now.