VISCOSITY NORTH AMERICA

Implementing Passkeys in Oracle APEX

Building a Passwordless Future

@kevintech

Kevin Herrarte

Software Architect

VISCOSITY NORTH AMERICA

Agenda

- Passkey Overview
  - What, Why, How
- Oracle APEX Implementation
- Jump into some code samples
- Demo
- Q&A

VISCOSITY
NORTH AMERICA

81%

of data breaches are caused by weak or stolen passwords

https://www.okta.com/identity-101/mistakes-that-lead-to-security-breach/

VISCOSITY
NORTH AMERICA

What are Passkeys?

One sentence: Public/private key pairs + local device storage

VISCOSITY
NORTH AMERICA

What Passkeys Fix

Secure, simple, phishing-resistant

VISCOSITY
NORTH AMERICA

Key Principles of Passkeys

No shared secrets

Device-bound credentials

VISCOSITY
NORTH AMERICA

How Passkeys Improve UX

Instant login with biometrics/FaceID/etc.

VISCOSITY NORTH AMERICA

Supported Browsers and Devices

- Chrome
  - Brave, Chromium, etc
- Safari
- Edge
- Firefox

- Android
- iOS
- Windows
- Mac

VISCOSITY NORTH AMERICA

How Passkeys Work?

WebAuthn API (W3C Standard)

Registration & Authentication Ceremonies

High-Level Flow

Registration

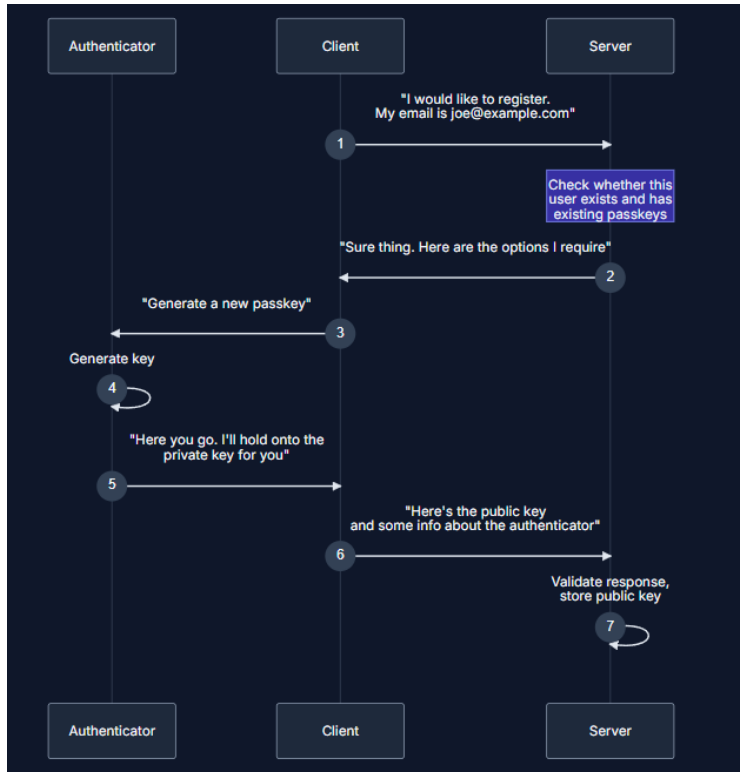
- Challenge → Device → Public key

Authentication

- Challenge → Private key sign → Server verifies

VISCOSITY NORTH AMERICA

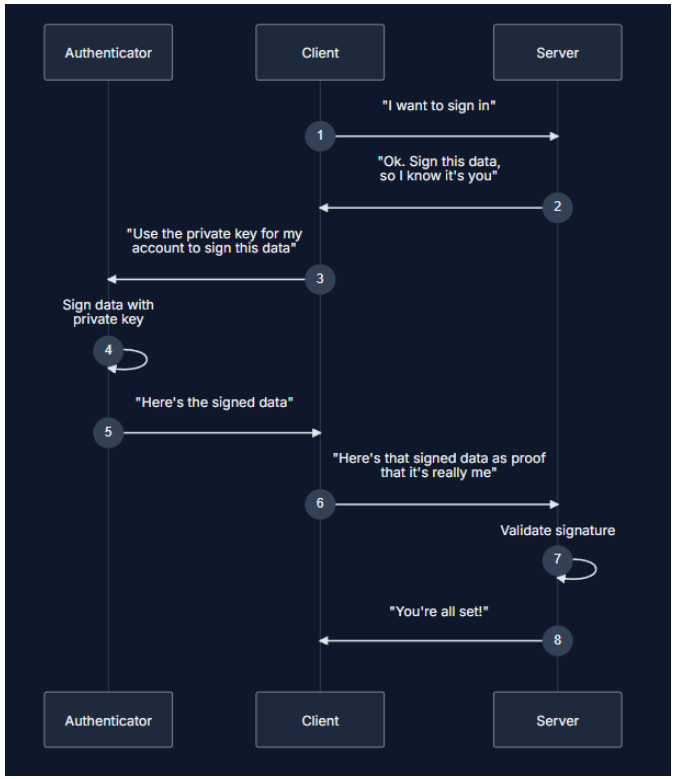
Ceremony: Registration Details



- User clicks “Register Passkey” in APEX page
- Client (JavaScript) calls PL/SQL AJAX to get a JSON “publicKeyOptions” object:
  - challenge (random bytes)
  - rp (relying party info)
  - user (user ID, name, displayName)
  - pubKeyCredParams (algorithms supported, e.g. -7 for ECDSA P-256)
- Browser’s navigator.credentials.create() invoked with publicKeyOptions
- Device shows biometric/PIN prompt, creates credential if user verifies
- Client sends attestationResponse to PL/SQL for validation and storage

VISCOSITY NORTH AMERICA

Ceremony: Authentication Details



- User clicks “Sign In with Passkey” on login page
- Client calls PL/SQL AJAX to get a JSON “publicKeyRequestOptions” object:
  - challenge (new random bytes)
  - allowCredentials (list of registered credentialId values)
  - timeout, userVerification flags
- Browser’s navigator.credentials.get() invoked with publicKeyRequestOptions
- Device prompts for biometric/PIN and returns assertion including signed data
- Client posts assertionResponse to PL/SQL to verify signature and create APEX session

VISCOSITY
NORTH AMERICA

Oracle APEX Implementation

VISCOSITY
NORTH AMERICA

What We Need in APEX?

WebAuthn on client + PL/SQL server logic

What We Need in APEX

- Database Table to store passkey credentials
  - **Columns:** credential_id, public_key, user_id
- Cryptographic package
  - AS_CRYPTO
- Backend
  - **PL/SQL** that handles ceremonies
- Frontend
  - **JavaScript** on APEX pages to invoke WebAuthn API
  - **Dynamic Actions** to trigger registration/authentication flows

Generating the Challenge

- Use AS_CRYPTO.RANDOMBYTES
- Convert to Base64 URL-encoded string for JSON payload
- Store challenge temporarily

```
l_challenge := AS_CRYPTO.RANDOMBYTES(32);
l_challenge_b := UTL_RAW.CAST_TO_VARCHAR2(UTL_ENCODE.BASE64_ENCODE(l_challenge));
```

VISCOSITY NORTH AMERICA

Returning the PublicKeyOptions (JSON)

- Build JSON server-side
- Return JSON to client via APEX's *apex_json* or CLOB directly

```
FUNCTION get_registration_options (p_user_id VARCHAR2) RETURN CLOB IS
  l_challenge   RAW(32);
  l_challenge_b VARCHAR2(100);
  l_json        CLOB;
BEGIN
  l_challenge := AS_CRYPTO.RANDOMBYTES(32);
  l_challenge_b := UTL_RAW.CAST_TO_VARCHAR2(UTL_ENCODE.BASE64_ENCODE(l_challenge));
  apex_util.set_session_state('P_REG_CHALLENGE', l_challenge_b);

  l_json := JSON_OBJECT(
    'challenge' VALUE l_challenge_b,
    'rp' VALUE JSON_OBJECT(
      'id' VALUE 'apex.oracle.com',
      'name' VALUE 'My APEX App'
    ),
    'user' VALUE JSON_OBJECT(
      'id' VALUE TO_BASE64(p_user_id),
      'name' VALUE p_user_id,
      'displayName' VALUE p_user_id
    ),
    'pubKeyCredParams' VALUE JSON_ARRAY(
      JSON_OBJECT('type' VALUE 'public-key','alg' VALUE -7)  -- ES256
    )
  ).to_clob;

  RETURN l_json;
END;
```

VISCOSITY NORTH AMERICA

Storing the Credential

- Client posts *attestationResponse* to an AJAX callback, e.g. *save_credential_response*
- Decode Base64 URL-encoded credentialId → RAW
- Verify that client *challenge* matches original *challenge*
- Verify signature using publicKey and Cryptographic package
- Save credentialId and publicKey into table

```
insert into apex_passkeys(user_id, credential_id, public_key)
values (:APP_USER, :CREDENTIAL_ID, :PUBLIC_KEY);
```

VISCOSITY NORTH AMERICA

Signature Verification with AS_CRYPTO

- Decode Base64 URL-encoded credentialId, authenticatorData, clientDataJSON, signature → RAW
- Reconstruct signedData = authenticatorData || HASH(clientDataJSON)
- Convert ASN.1 ECDSA signature to raw format (concatenated R + S values).
- Use AS_CRYPTO.VERIFY with stored public_key to validate signature over signedData
- Create APEX session manually

```
as_crypto.verify ( p_signed_data -- expected value (original msg)
                  ,convert_asn1_to_raw(base64url_decode(p_signature)) -- signature
                  ,utl_raw.cast_to_raw(base64url_to_base64(p_public_key)) -- encoded public key
                  ,as_crypto.KEY_TYPE_EC -- key algo
                  ,as_crypto.SIGN_SHA256withECDSAinP1363 -- key algo
                  );
```

VISCOSITY NORTH AMERICA

Retrieving Credentials for Authentication

- Query *apex_passkeys* for this user_id → fetch credential_id
- Generate a new challenge (random bytes)
- Build JSON:

```
{
  "challenge": "Base64Url(...)",
  "allowCredentials": [
    {
      "id": "Base64Url(credential_id)",
      "type": "public-key"
    }
  ],
  "userVerification": "preferred"
}
```

VISCOSITY NORTH AMERICA

Building UX in APEX

- Registration
  - On your Profile page, an ‘Enable Passkey’ button triggers registration.
  - Dynamic Action → *navigator.credentials.create()* with *publicKeyOptions*
  - On success → PL/SQL *save_credential_response* with *attestationResponse*
- Authentication
  - On your Login page, a ‘Sign in with Passkey’ button triggers authentication.
  - Dynamic Action → *navigator.credentials.get()* → post assertion to PL/SQL *verify_assertion_response*
  - On success, redirect to home page; on failure, show error message region

VISCOSITY NORTH AMERICA

Building UX in APEX

```
apex.server.process( "ENROLL", {
  pageItems: "#P1_USERNAME"
}, {
  success: async function( data )  {
    const credential = await navigator.credentials.create({
      publicKey: { // publicKeyOptions JSON
        challenge: new Uint8Array(data.challenge), // base64url-encoded
        rp: {
          id: data.rpId,
          name: "My APEX Application"
        },
        user: {
          id: new Uint8Array(data.userId), // base64url-encoded
          name: "John Doe",
          displayName: "John Doe"
        },
        pubKeyCredParams: [
          {
            alg: -7, // ES256
            type: "public-key"
          }
        ],
      }
    });

    // send credential to server ...
  },
  error: function( jqXHR, textStatus, errorThrown ) {
    // handle error
  }
} );
```

```
apex.server.process( "AUTHENTICATION", {
  pageItems: "#P1_USERNAME"
}, {
  success: async function( data )  {
    const assertion = await navigator.credentials.get({
      publicKey: {
        challenge: data.challenge, // challenge from server base64 url-encoded
        allowCredentials: data.allowCredentials, // array of credentials
        timeout: 60000,
        userVerification: "preferred",
      }
    });

    // send assertion to server ...
  },
  error: function( jqXHR, textStatus, errorThrown ) {
    // handle error
  }
} );
```

VISCOSITY NORTH AMERICA

Managing Sessions Post-Authentication

- Set APEX session context manually using *APEX_AUTHENTICATION.POST_LOGIN*
- Redirect user to “Home” page

```
APEX_AUTHENTICATION.POST_LOGIN(
    p_username => l_username,
    p_password => 'fake password',
    p_uppercase_username => TRUE
  );
```

Optional: Check Enrollment Status

- Show biometric button only if enrolled
- In the login page's "On Load" process:
  - Query apex_passkeys table for user_id = :APP_USER
  - If row exists → enable "Sign in with Passkey" button (display icon)
  - If no row → hide passkey button
- Dynamic Action can show/hide APEX region or button based on SQL condition

Where Things Can Break

- Challenge Mismatch:
  - Dev must store and compare the exact challenge (little/big-endian issues)
- Origin Mismatch:
  - WebAuthn requires exact “origin” (scheme + hostname + port) match. Test in dev URL vs prod.
- Encoding/Decoding Errors:
  - Extracting publicKey from attestation requires correct parsing.
- Browser/Device Compatibility:
  - Some older browser versions may not fully support WebAuthn.

VISCOSITY NORTH AMERICA

Handling Errors Gracefully

- On Client Side (JavaScript):
  - Capture exceptions from navigator.credentials.create/get()
  - Show user-friendly message: “Passkey authentication failed. Please try again.”
- On Server Side (PL/SQL):
  - Wrap signature verification in BEGIN … EXCEPTION to return clear JSON error code
- UX Tips:
  - Provide fallback to “Login with password” if passkey fails repeatedly
  - Log exceptions in APEX debug logs to troubleshoot

VISCOSITY
NORTH AMERICA

We have a Plug-In for that…

Plugin Introduction

VISCOSITY
NORTH AMERICA

Challenges of Manual Implementation

Complexity in handling edge cases

VISCOSITY
NORTH AMERICA

Why We Built the Plugin

Simplify life for APEX developers

VISCOSITY
NORTH AMERICA

What the Plugin Automates

Enrollment, Authentication, Silent Checks

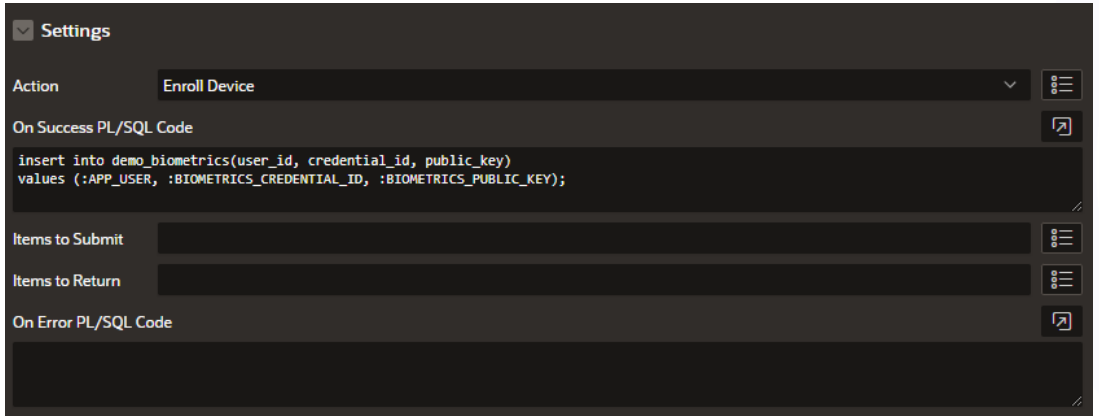
VISCOSITY
NORTH AMERICA

How to Use the Plugin

Simple configuration & Dynamic Actions

VISCOSITY NORTH AMERICA

Real Example: Plugin Setup



VISCOSITY
NORTH AMERICA

Live Demo

VISCOSITY
NORTH AMERICA

Summary of Key Takeaways

VISCOSITY
NORTH AMERICA

Try it out, integrate it, go passwordless

Next Steps

Any Questions?

Kevin.Herrarte@ViscosityNA.com

@kevintech

VISCOSITY
NORTH AMERICA

VISCOSITY
NORTH AMERICA

Where to Get the Plugin?

Scan QR Code to get the Code + Documentation

https://linktr.ee/kevintech.ninja

VISCOSITY NORTH AMERICA

Thank You!

- Facebook.com/ViscosityNA
- LinkedIn.com/company/Viscosity-North-America
- @ViscosityNA
- Viscosity North America
- @Viscosity_NA