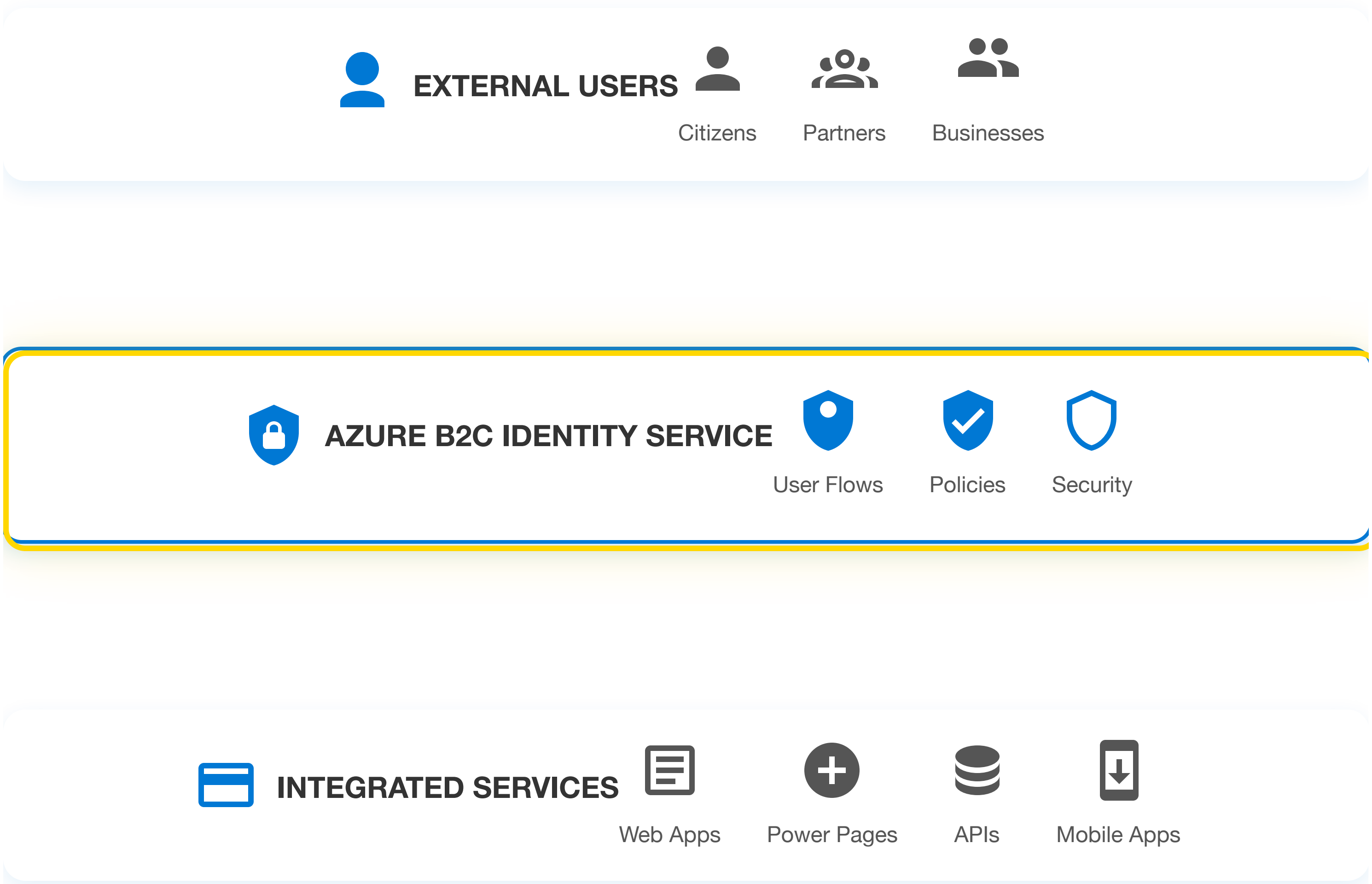
Azure B2C - Secure Enterprise SSO for {ORG NAME}

Architecture, Implementation & Operational Guide

PBMM GUIDELINES



EXTERNAL USERS

Citizens

Partners

Businesses

AZURE B2C IDENTITY SERVICE

User Flows

Policies

Security

INTEGRATED SERVICES

Web Apps

Power Pages

APIs

Mobile Apps

Protected B Compliant

Canadian Data Residency

ITSG-33 Compliant

Multi-Factor Authentication

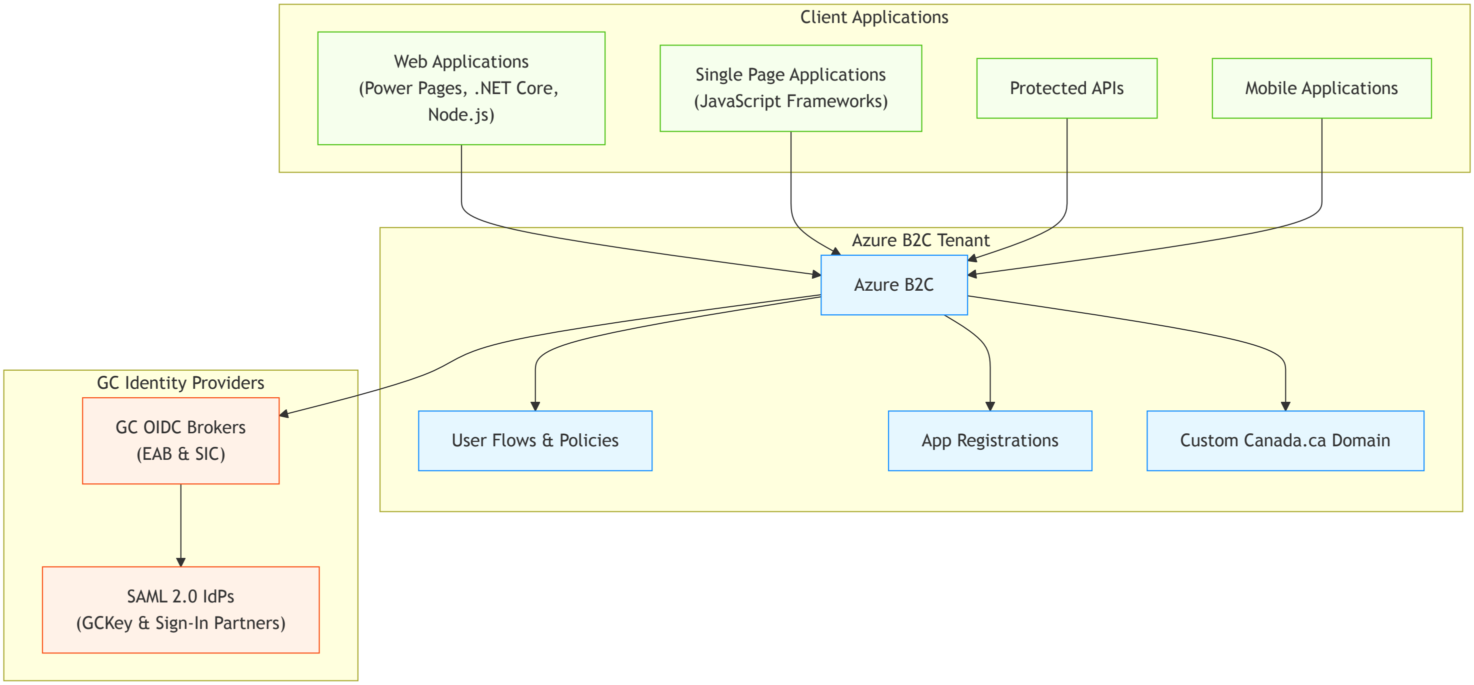
GCKey & Sign-In Partners

EXECUTIVE OVERVIEW

This document defines the architecture, implementation, and operational requirements for enabling secure single sign-on (SSO) services using Azure B2C for the Government of Canada department's web applications and APIs requiring Protected B compliance. Azure B2C provides business-to-customer identity as a service that adheres to Government of Canada security standards including ITSG-33, ISO 27001, and Protected B requirements.

The implementation will:

1. Provide a secure enterprise identity platform for external users accessing departmental services
2. Enable integration with Government of Canada identity services (GCKey/Sign-in Partner via OIDC brokers)
3. Implement Protected B security posture meeting all Government of Canada cloud security requirements
4. Support multiple platforms including Power Pages portals, .NET applications, Node.js, and Python
5. Deliver both web application SSO (OpenID Connect) and API authorization (OAuth 2.0)
6. Establish a standardized application registration process with proper governance
7. Create a scalable service management framework for enterprise-wide operations


Client Applications
Web Applications (Power Pages, .NET Core, Node.js)
Single Page Applications (JavaScript Frameworks)
Protected APIs
Mobile Applications
Azure B2C Tenant
Azure B2C
User Flows & Policies
App Registrations
Custom Canada.ca Domain
GC Identity Providers
GC OIDC Brokers (EAB & SIC)
SAML 2.0 IdPs (GCKey & Sign-In Partners)

ALIGNMENT TO GC CLOUD USAGE PROFILES AND CONNECTION PATTERNS

This implementation aligns with the Government of Canada's Cloud Adoption Strategy and follows approved cloud usage profiles and connection patterns documented in the GC Cloud Security Control Framework.

Applicable Connection Patterns

Reference	Scenario	Application to Azure B2C
C	External user access to cloud-based service	Primary pattern: Non-GC users accessing departmental services via Azure B2C authentication
D	Service/Application Interoperability	API-to-API authentication using OAuth 2.0 token-based flows
E	Cloud Administration and Management	Administrative access to the Azure B2C tenant for configuration and monitoring

Cloud Usage Profiles

This implementation aligns with GC Cloud Profile 4 (Protected B / Medium Integrity / Medium Availability) for the production environment, with development environments aligned to Profile 1 or 2 as appropriate. By deploying Azure B2C as a SAAS solution, we leverage Microsoft's robust security model while implementing additional department-specific guardrails to ensure full compliance with Protected B requirements.

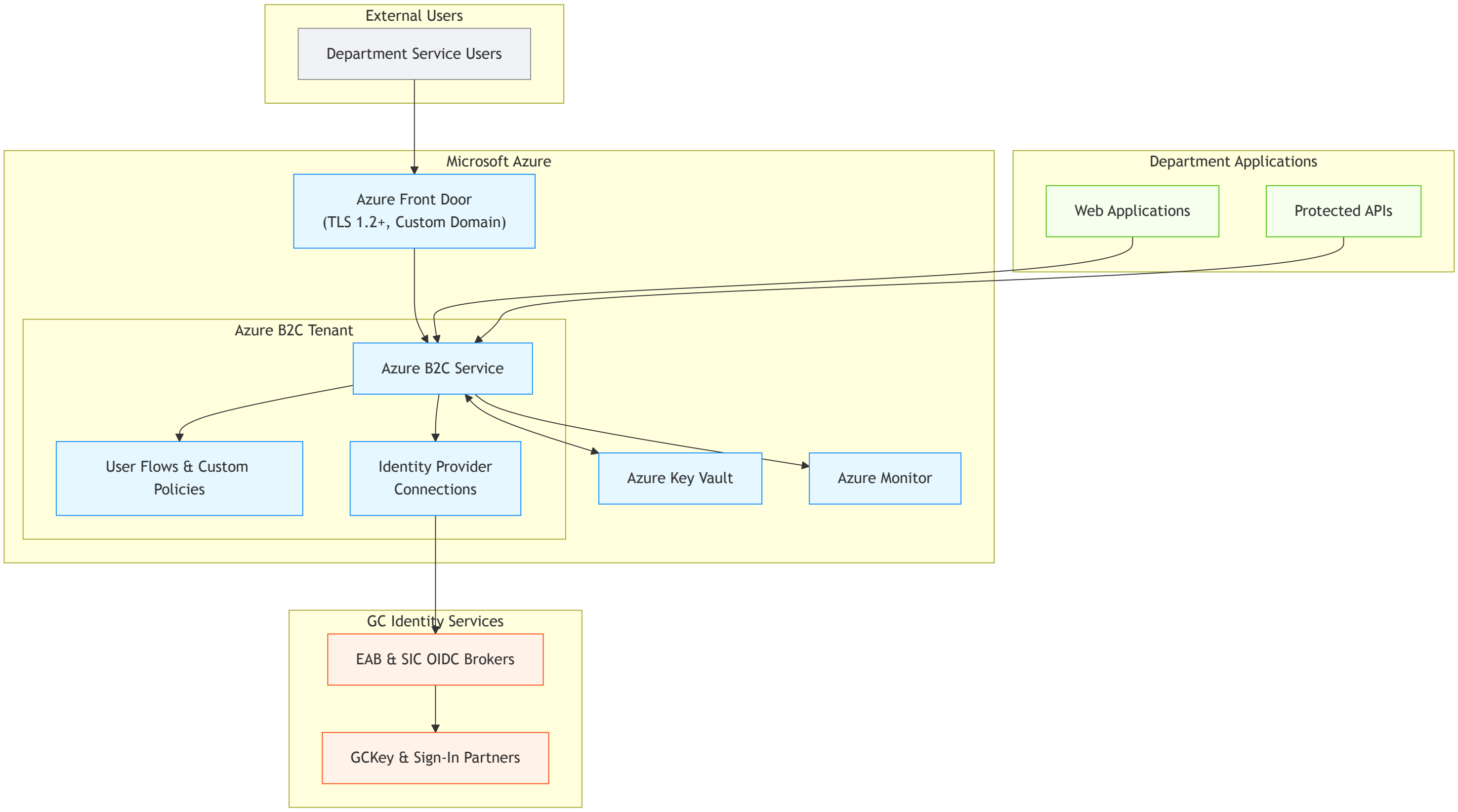
IMPLEMENTATION ARCHITECTURE COMPONENTS

Azure B2C Core Architecture

The department's implementation will utilize the following key components:

1. **Azure B2C Tenant**: A dedicated B2C directory separate from the department's primary Azure AD
2. **Azure Front Door**: CDN service to apply TLS 1.2+ requirements and provide custom domain capabilities
3. **Key Vault**: Secure storage for certificates, secrets and encryption keys

4. **Azure Monitor**: Comprehensive logging and alerting for security compliance
5. **App Registrations**: Secured application identities for each integrated service
6. **User Flows**: Customized authentication workflows for various application types
7. **Custom Policies**: For advanced scenarios requiring greater control over the authentication processes
8. **Custom Domain**: Canada.ca subdomain for the user experience



Integration with GC Identity Services

The department's Azure B2C implementation will integrate with the Government of Canada's identity ecosystem through OpenID Connect brokers:

1. **Enterprise Authorization Broker (EAB)**: SSC's OIDC broker service for GCKey and Sign-in Partners
2. **Sign-in Canada (SIC)**: TBS's OIDC broker service for GCKey and Sign-in Partners

Rather than implementing complex SAML 2.0 integrations directly with GCKey/Sign-in Partners, Azure B2C will connect to these OIDC brokers which handle the SAML complexity on the department's behalf, including the SOAP binding required for proper single logout that B2C does not natively support.

SECURITY AND COMPLIANCE IMPLEMENTATION

Protected B Compliance Requirements

The Azure B2C implementation must adhere to the ITSG-33 control profile for Protected B information, which includes:

1. **Data Protection**: Encryption in transit (TLS 1.2+) and at rest (AES-256)
2. **Identity and Access Management**: MFA, strong authentication, privileged access controls
3. **Boundary Defense**: Network segmentation, DLP controls, WAF protection
4. **Monitoring and Logging**: Comprehensive audit trails for user activities
5. **Incident Response**: Detection and response capabilities for security events

Guardrails Implementation

The following guardrails must be implemented to achieve Protected B compliance:

Category	Guardrail	Implementation Approach
Access Control	Minimum of 2 Global Admins with MFA	Configure department-federated accounts with appropriate privileges
Access Control	Application segmentation	Create dedicated security groups for each integrated application
Network Security	TLS 1.2+ enforcement	Deploy Azure Front Door to enforce TLS version
Network Security	Custom Canada.ca domain	Deploy Azure Front Door with Canada.ca subdomain
Cryptography	Use Entrust certificates	Generate CSRs from Key Vault for all TLS endpoints
Cryptography	RSA 2048+ for tenant keys	Configure B2C encryption settings to use strong cryptography
Authentication	JWT signatures for federation	Use Entrust certificates for token signing
Authentication	Prohibit implicit flows	Standardize on authorization code flow with PKCE

Category	Guardrail	Implementation Approach
Secret Management	Centralized secret storage	Store all credentials in Azure Key Vault
Session Management	Front-channel logout	Require implementation for all applications
Monitoring	Risk detection enabled	Configure risk detection policies in Azure AD B2C Premium P2
Geography	Canadian region selection	Ensure Canadian region selection for data residency
Environment	Dev/Test/Prod separation	Maintain separate environments with consistent controls
Audit	Log retention	Configure Azure Monitor for 1+ year retention
User Experience	Track user behavior	Implement Application Insights for user journey analysis

BUILD BOOK: DETAILED IMPLEMENTATION STEPS

A. PROVISIONING AZURE B2C TENANT

A.1 Prerequisites

- Global Administrator rights to the Azure subscription
- Separate subscriptions for Development and Production environments
- Access to Azure Key Vault for certificate and secret management
- Access to DNS for Canada.ca domain configuration

A.2 Tenant Creation Process

1. Sign into the Azure portal as Global Administrator
2. Navigate to "Create a resource" > search for "Azure Active Directory B2C"
3. Select "Create a new Azure AD B2C Tenant"
4. Configure tenant with appropriate naming:
    - Organization name: `{DEPARTMENT}-b2c-{ENV}`
    - Initial domain name: `{DEPARTMENT}b2c{ENV}`

    - Country: Canada
    - Subscription: Select appropriate subscription
    - Resource group: Create dedicated RG for B2C resources
    - Resource group region: Canada Central or Canada East
5. Complete validation and create the tenant
6. Switch to the new B2C tenant directory
7. Add additional administrators as Global Administrators to the tenant:
    - Select "Users" from the Azure AD B2C menu
    - Create new invited users with Global Administrator role
    - Ensure all administrators use MFA

A.3 Tenant Branding Configuration

1. Navigate to "Company Branding" in the Azure B2C tenant
2. Configure the default brand with:
    - Sign-in page background image: Government of Canada banner
    - Banner logo: Canada wordmark
    - Configure additional elements to match GC design system
3. Ensure all branding elements meet official GC visual identity requirements

B. CONFIGURING USER AUTHENTICATION FLOWS

B.1 Local Account User Flow Configuration

1. Navigate to "User flows" in the Azure B2C tenant
2. Create a new "Sign up and sign in" user flow:
    - Select "Recommended" version
    - Name: `{APPNAME}_susi`
    - Identity providers: Email signup
    - MFA: Conditional (based on risk)
    - Collect attributes: Email, Given Name, Surname
    - Return claims: Minimum necessary for application function
3. Configure user flow properties:
    - Enable "Require ID Token in Logout Requests"
    - Configure page layouts to match GC design system
    - Set appropriate session behavior
4. Create additional flows as needed:
    - Profile editing: `{APPNAME}_editprofile`
    - Password reset: `{APPNAME}_pwdreset`

B.2 Enterprise Access Broker (EAB) Integration

1. Navigate to "Identity providers" in the Azure B2C tenant
2. Create a new "OpenID Connect" provider:
    - Name: Enterprise Access Broker
    - Metadata URL: `{EAB_METADATA_URL}`
    - Client ID: Provided by SSC
    - Client secret: Provided by SSC
    - Scope: openid
    - Response type: code
    - Response mode: query
    - User ID: sub
    - Display name: sub
3. Create a dedicated user flow for EAB:
    - Name: `eab_sso`
    - Identity providers: Select only EAB
    - Configure appropriate claim mapping
4. Submit Azure B2C metadata and redirect URLs to SSC for configuration:
    - Tenant ID: Copy from B2C properties
    - Metadata URL: Copy from user flow "Run" screen
    - Redirect URL: Copy from user flow "Run" screen

B.3 Sign-in Canada (SIC) Integration

1. Navigate to "Identity providers" in the Azure B2C tenant
2. Create a new "OpenID Connect" provider:
    - Name: Sign-in Canada
    - Metadata URL: `{SIC_METADATA_URL}`
    - Client ID: Provided by TBS
    - Client secret: Provided by TBS
    - Scope: openid
    - Response type: code
    - Response mode: query
    - User ID: sub
    - Display name: sub
3. Create a dedicated user flow for SIC:
    - Name: `sic_sso`
    - Identity providers: Select only SIC
    - Configure appropriate claim mapping

4. Submit Azure B2C metadata and redirect URLs to TBS for configuration:
    - Tenant ID: Copy from B2C properties
    - Metadata URL: Copy from user flow "Run" screen
    - Redirect URL: Copy from user flow "Run" screen

C. IMPLEMENTING CUSTOM DOMAIN AND TLS ENFORCEMENT

C.1 Custom Domain Registration

1. Register custom domain in Azure AD B2C:
    - Navigate to "Custom domain names"
    - Add custom domain (e.g., `auth.{DEPARTMENT}.canada.ca`)
2. Submit DNS change request to SSC for TXT verification:
    - Domain: `auth.{DEPARTMENT}.canada.ca`
    - Type: TXT
    - Value: Microsoft verification string
3. Verify domain in Azure AD B2C after DNS propagation

C.2 Azure Front Door Deployment (Recommended Approach)

1. Create Azure Front Door resource:

```powershell
# Create Resource Group if it doesn't exist
New-AzResourceGroup -Name "rg-{DEPARTMENT}-afd-{ENV}" -Location "Canada Central"

# Create Front Door profile
New-AzFrontDoorCdnProfile -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -Name "afd-{DEPARTMENT}-b2c-{ENV}" -Location "Global" -Sku "Standard_AzureFrontDoor

# Create endpoint
New-AzFrontDoorCdnEndpoint -EndpointName "auth-{DEPARTMENT}-{ENV}" `
  -ProfileName "afd-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -Location "Global"

# Add origin group
New-AzFrontDoorCdnOriginGroup -OriginGroupName "b2c-origin-group" `
  -ProfileName "afd-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -HealthProbePath "/" -HealthProbeRequestType "HEAD" `
  -HealthProbeProtocol "Https" -SessionAffinityState "Disabled"

# Add B2C origin
New-AzFrontDoorCdnOrigin -OriginName "b2c-origin" `
  -OriginGroupName "b2c-origin-group" `
  -ProfileName "afd-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -HostName "{TENANT}.b2clogin.com" `
  -OriginHostHeader "{TENANT}.b2clogin.com" `
  -HttpPort 80 -HttpsPort 443 -Priority 1 -Weight 1000 -Enabled $true

# Create route
New-AzFrontDoorCdnRoute -RouteName "b2c-route" `
  -EndpointName "auth-{DEPARTMENT}-{ENV}" `
  -ProfileName "afd-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -OriginGroupId "/subscriptions/{SUBSCRIPTION_ID}/resourceGroups/rg-{DEPARTMENT}-afd
  -SupportedProtocols @("Http", "Https") `
  -PatternsToMatch @("/*") -ForwardingProtocol "HttpsOnly" `
  -LinkToDefaultDomain "Enabled" -HttpsRedirect "Enabled" `
  -EnabledState "Enabled"
```

2. Generate Entrust certificate for custom domain:
    - Create Key Vault CSR for `auth.{DEPARTMENT}.canada.ca`

```
# Create CSR in Key Vault
$policy = New-AzKeyVaultCertificatePolicy -SecretContentType "application/x-pkcs12" `
  -SubjectName "CN=auth.{DEPARTMENT}.canada.ca" `
  -IssuerName "Unknown" `
  -ValidityInMonths 12 `
  -ReuseKeyOnRenewal `
  -KeyType "RSA" `
  -KeySize 2048 `
  -KeyUsage "DigitalSignature", "KeyEncipherment" `
  -Ekus @("1.3.6.1.5.5.7.3.1", "1.3.6.1.5.5.7.3.2") `
  -CertificateTransparency $true

Add-AzKeyVaultCertificate -VaultName "kv-{DEPARTMENT}-b2c-{ENV}" `
  -Name "auth-{DEPARTMENT}-canada-ca" `
  -CertificatePolicy $policy
```

- Download CSR from Key Vault and submit to Entrust CA
- After receiving the certificate, import to Key Vault:

```
# Merge certificate with CSR in Key Vault
Import-AzKeyVaultCertificateMerge -VaultName "kv-{DEPARTMENT}-b2c-{ENV}" `
  -Name "auth-{DEPARTMENT}-canada-ca" `
  -FilePath "path/to/certificate.cer"
```

3. Configure AFD with custom domain certificate:

```
# Create custom domain in AFD
New-AzFrontDoorCdnCustomDomain -CustomDomainName "auth-{DEPARTMENT}-canada-ca" `
  -HostName "auth.{DEPARTMENT}.canada.ca" `
  -ProfileName "afd-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -MinimumTlsVersion "TLS12" `
  -CertificateSource "AzureKeyVault" `
  -SecretId "/subscriptions/{SUBSCRIPTION_ID}/resourceGroups/rg-{DEPARTMENT}-b2c-{ENV

# Update route to use custom domain
Update-AzFrontDoorCdnRoute -RouteName "b2c-route" `
  -EndpointName "auth-{DEPARTMENT}-{ENV}" `
  -ProfileName "afd-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -CustomDomainId "/subscriptions/{SUBSCRIPTION_ID}/resourceGroups/rg-{DEPARTMENT}-af
```

4. Configure TLS 1.2+ enforcement:

```
# Create security policy to enforce TLS 1.2+
New-AzFrontDoorCdnSecurityPolicy -ProfileName "afd-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -SecurityPolicyName "tls-policy" `
  -SecurityType "SecurityPolicies" `
  -MinimumTlsVersion "TLS12" `
  -EndpointName "auth-{DEPARTMENT}-{ENV}"
```

5. Setup WAF policy (Optional but recommended):

```
# Create WAF policy
New-AzFrontDoorWafPolicy -Name "waf-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -Sku "Standard_AzureFrontDoor" `
  -Mode "Prevention" `
  -EnabledState "Enabled" `
  -CustomBlockResponseStatusCode 403

# Add managed ruleset
New-AzFrontDoorWafManagedRuleObject -Type "DefaultRuleSet" `
  -Version "1.0" `
  -Action "Block" `
  -WafPolicy $wafPolicy

# Apply WAF policy to security policy
New-AzFrontDoorCdnSecurityPolicy -ProfileName "afd-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -SecurityPolicyName "waf-policy" `
  -SecurityType "WebApplicationFirewall" `
  -WafPolicyId "/subscriptions/{SUBSCRIPTION_ID}/resourceGroups/rg-{DEPARTMENT}-afd-{
  -EndpointName "auth-{DEPARTMENT}-{ENV}"
```

C.3 Azure Application Gateway Deployment (Alternative Approach)

Azure Application Gateway can be used as an alternative to Azure Front Door when:

- You require more advanced WAF capabilities
- You need additional network-level control
- You prefer a regional deployment instead of global
- You need to integrate with on-premises networks via ExpressRoute

1. Create Virtual Network for Application Gateway:

```
# Create VNET and subnet
New-AzVirtualNetwork -ResourceGroupName "rg-{DEPARTMENT}-appgw-{ENV}" `
  -Location "Canada Central" `
  -Name "vnet-{DEPARTMENT}-appgw-{ENV}" `
  -AddressPrefix "10.0.0.0/16"

Add-AzVirtualNetworkSubnetConfig -Name "snet-appgw" `
  -VirtualNetwork $vnet `
  -AddressPrefix "10.0.1.0/24"

$vnet | Set-AzVirtualNetwork
```

2. Create public IP for Application Gateway:

```
New-AzPublicIpAddress -ResourceGroupName "rg-{DEPARTMENT}-appgw-{ENV}" `
  -Location "Canada Central" `
  -Name "pip-{DEPARTMENT}-appgw-{ENV}" `
  -AllocationMethod "Static" `
  -Sku "Standard"
```

3. Create Application Gateway configuration:

```
# Create IP configuration
$vnet = Get-AzVirtualNetwork -Name "vnet-{DEPARTMENT}-appgw-{ENV}" -ResourceGroupName
$subnet = Get-AzVirtualNetworkSubnetConfig -Name "snet-appgw" -VirtualNetwork $vnet
$pip = Get-AzPublicIpAddress -Name "pip-{DEPARTMENT}-appgw-{ENV}" -ResourceGroupName

$gipconfig = New-AzApplicationGatewayIPConfiguration -Name "appgw-ip-config" -Subnet

# Create frontend port
$frontendport = New-AzApplicationGatewayFrontendPort -Name "appgw-frontend-port" -Por

# Create frontend IP config
$frontendip = New-AzApplicationGatewayFrontendIPConfig -Name "appgw-frontend-ip" -Pub

# Create backend pool
$backendPool = New-AzApplicationGatewayBackendAddressPool -Name "b2c-backend-pool" `
  -BackendIPAddresses "{TENANT}.b2clogin.com"

# Create backend settings
$backendsetting = New-AzApplicationGatewayBackendHttpSetting -Name "b2c-backend-setti
  -Port 443 `
  -Protocol "Https" `
  -CookieBasedAffinity "Disabled" `
  -HostName "{TENANT}.b2clogin.com" `
  -RequestTimeout 30

# Create certificate from Key Vault
$sslCert = New-AzApplicationGatewaySslCertificate -Name "b2c-ssl-cert" `
  -KeyVaultSecretId "https://kv-{DEPARTMENT}-b2c-{ENV}.vault.azure.net/secrets/auth-{

# Create HTTP listener
$listener = New-AzApplicationGatewayHttpListener -Name "b2c-https-listener" `
  -Protocol "Https" `
  -FrontendIPConfiguration $frontendip `
  -FrontendPort $frontendport `
  -SslCertificate $sslCert `
  -HostName "auth.{DEPARTMENT}.canada.ca"

# Create routing rule
$rule = New-AzApplicationGatewayRequestRoutingRule -Name "b2c-routing-rule" `
  -RuleType "Basic" `
  -HttpListener $listener `
  -BackendAddressPool $backendPool `
  -BackendHttpSettings $backendsetting
```

```
# Create WAF configuration
$wafConfig = New-AzApplicationGatewayWebApplicationFirewallConfiguration `
  -Enabled $true `
  -FirewallMode "Prevention" `
  -RuleSetType "OWASP" `
  -RuleSetVersion "3.2"

# Create SSL policy to enforce TLS 1.2+
$sslPolicy = New-AzApplicationGatewaySslPolicy -PolicyType "Custom" `
  -MinProtocolVersion "TLSv1_2" `
  -CipherSuite @("TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384",
                "TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256",
                "TLS_DHE_RSA_WITH_AES_256_GCM_SHA384",
                "TLS_DHE_RSA_WITH_AES_128_GCM_SHA256")

# Create Application Gateway with WAF v2 SKU
New-AzApplicationGateway -Name "appgw-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-appgw-{ENV}" `
  -Location "Canada Central" `
  -BackendAddressPools $backendPool `
  -BackendHttpSettingsCollection $backendsetting `
  -FrontendIPConfigurations $frontendip `
  -GatewayIPConfigurations $gipconfig `
  -FrontendPorts $frontendport `
  -HttpListeners $listener `
  -RequestRoutingRules $rule `
  -Sku @{Name = "WAF_v2"; Tier = "WAF_v2"; Capacity = 2} `
  -WebApplicationFirewallConfig $wafConfig `
  -SslCertificates $sslCert `
  -SslPolicy $sslPolicy
```

4. Submit DNS change request to SSC for your custom domain:
   - Domain: `auth.{DEPARTMENT}.canada.ca`
   - Type: A/CNAME
   - Value: Public IP address or DNS name of Application Gateway
5. Configure health probe for B2C endpoints:

```
$probe = New-AzApplicationGatewayProbeConfig -Name "b2c-health-probe" `
  -Protocol "Https" `
  -Path "/" `
  -Interval 30 `
  -Timeout 30 `
  -UnhealthyThreshold 3 `
  -PickHostNameFromBackendHttpSettings $false `
  -Host "{TENANT}.b2clogin.com"

$appgw = Get-AzApplicationGateway -Name "appgw-{DEPARTMENT}-b2c-{ENV}" -ResourceGroup
$appgw = Add-AzApplicationGatewayProbeConfig -ApplicationGateway $appgw -Name "b2c-he
$appgw = Set-AzApplicationGateway -ApplicationGateway $appgw
```

C.4 URL Redirect Configuration

1. Configure B2C to redirect to custom domain when accessed directly:
   - In Azure Portal, navigate to Azure AD B2C tenant
   - Select "Properties"
   - Under "Domain name for branding", enter your custom domain
   - Save the configuration
2. Test all authentication flows to ensure proper domain usage:
   - Verify redirects work correctly
   - Ensure TLS 1.2+ is enforced
   - Validate certificate trust chain
3. Implement URL rewrite rules:

   For Azure Front Door:

```
# Create a rule to handle B2C paths correctly
New-AzFrontDoorCdnRuleSet -ProfileName "afd-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -RuleSetName "b2c-rules"

New-AzFrontDoorCdnRule -ProfileName "afd-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -RuleSetName "b2c-rules" `
  -RuleName "rewrite-tenant-domain" `
  -Order 1 `
  -Condition @{MatchVariable="RequestUri"; Operator="Contains"; Value="{TENANT}.onmic
  -Action @{Type="UrlRewrite"; SourcePattern="{TENANT}.onmicrosoft.com"; DestinationP

# Associate rule set with route
Update-AzFrontDoorCdnRoute -RouteName "b2c-route" `
  -EndpointName "auth-{DEPARTMENT}-{ENV}" `
  -ProfileName "afd-{DEPARTMENT}-b2c-{ENV}" `
  -ResourceGroupName "rg-{DEPARTMENT}-afd-{ENV}" `
  -RuleSetId "/subscriptions/{SUBSCRIPTION_ID}/resourceGroups/rg-{DEPARTMENT}-afd-{EN
```

For Application Gateway:

```
# Create URL rewrite configuration
$rewriteRuleSet = New-AzApplicationGatewayRewriteRuleSet -Name "b2c-rewrite-ruleset"

$rewriteRule = New-AzApplicationGatewayRewriteRule -Name "rewrite-tenant-domain" `
  -ActionSet @{
    UrlConfiguration = @{
      ModifiedUrl = @{
        Value = "https://auth.{DEPARTMENT}.canada.ca{http_req_url}"
      }
    }
  } `
  -Condition @{
    Variable = "http_req_url"
    Pattern = "{TENANT}.onmicrosoft.com"
    IgnoreCase = $true
    Negate = $false
  }

$rewriteRuleSet.RewriteRules.Add($rewriteRule)
$appgw = Add-AzApplicationGatewayRewriteRuleSet -ApplicationGateway $appgw -Name "b2c
$appgw = Set-AzApplicationGateway -ApplicationGateway $appgw
```

4. Block direct access to B2C tenant:
    - Create WAF rules to block requests that bypass your custom domain
    - Monitor for any direct access attempts to the B2C tenant
    - Implement IP restrictions if necessary to limit administrative access

D. APPLICATION INTEGRATION PATTERNS

D.1 Web Application Integration (OpenID Connect)

1. Create App Registration for the web application:
    - Name: `{APPNAME}-{ENV}`
    - Supported account types: Accounts in this organizational directory only
    - Redirect URI: Application callback URL
    - Application ID URI: Generated automatically
2. Configure authentication settings:
    - Platform configurations: Web
    - Redirect URIs: Application callback URLs
    - Front-channel logout URL: Application logout endpoint
    - Implicit grant: Do NOT enable ID tokens

3. Configure application settings:
    - Client secret: Generate and store in Key Vault
    - API permissions: None required for basic authentication
4. Provide integration details to application team:
    - Client ID: App registration client ID
    - Client secret: Key Vault reference
    - Authority: `https://auth.{DEPARTMENT}.canada.ca/{TENANT}/{USERFLOW}`
    - Metadata URL: User flow metadata endpoint

D.2 Single Page Application Integration

1. Create App Registration for SPA:
    - Name: `{APPNAME}-{ENV}`
    - Supported account types: Accounts in this organizational directory only
    - Redirect URI: SPA application URL
2. Configure authentication settings:
    - Platform configurations: Single-page application
    - Redirect URIs: SPA application URLs
    - Access tokens and authorization codes
    - Do NOT enable implicit flow for ID tokens
3. Configure CORS settings for SPA domains
4. Provide integration details to application team:
    - Client ID: App registration client ID
    - Authority: `https://auth.{DEPARTMENT}.canada.ca/{TENANT}/{USERFLOW}`
    - Metadata URL: User flow metadata endpoint
    - Scopes: Authorize appropriate scopes

D.3 API Integration (OAuth 2.0)

1. Create App Registration for the API:
    - Name: `{APINAME}-{ENV}`
    - Supported account types: Accounts in this organizational directory only
2. Configure Expose an API settings:
    - Application ID URI: Set custom URI if needed
    - Add scopes: Define appropriate scopes with descriptive names
3. Create App Registration for client application:
    - Configure API permissions to the API scopes
    - Generate client secret or certificate
4. Provide integration details to application teams:

- Authority: `https://auth.{DEPARTMENT}.canada.ca/{TENANT}`
- Token endpoint: `/oauth2/v2.0/token`
- Client ID and secret/certificate references
- Authorized scopes

D.4 Power Pages Portal Integration

1. Create App Registration for Power Pages portal:
    - Name: `{PORTALNAME}-{ENV}`
    - Redirect URI: Power Pages authentication callback URL
    - Front-channel logout URL: Portal logout URL
2. Configure portal authentication in Power Pages admin:
    - Authentication type: OpenID Connect
    - Display Name: GC Single Sign-On
    - Client ID: App registration client ID
    - Client secret: From Key Vault
    - Authority: `https://auth.{DEPARTMENT}.canada.ca/{TENANT}/{USERFLOW}`
    - Metadata URL: User flow metadata endpoint
3. Configure logout URL in portal site settings
4. Test authentication flow and verify claims mapping

E. MONITORING AND SECURITY OPERATIONS

E.1 Logging and Monitoring Setup

1. Create Log Analytics workspace for Azure B2C logs:
    - Deploy in Canadian region
    - Configure appropriate retention period (minimum 1 year)
2. Configure diagnostic settings in Azure B2C:
    - Send audit logs to Log Analytics
    - Send sign-in logs to Log Analytics
    - Configure streaming to Event Hub if required for SIEM integration
3. Deploy Application Insights for user journey tracking:
    - Configure custom events for authentication flows
    - Implement user behavior analytics

E.2 Security Alerts Configuration

1. Configure Azure Monitor alerts for:
    - Failed authentication attempts exceeding threshold

    - Administrative actions on tenant
    - Risk detection events
    - Unusual sign-in patterns
2. Create action groups for alert notifications:
    - Email to security team
    - Integration with departmental ITSM system
    - Optional SMS for critical alerts
3. Implement custom KQL queries for advanced detection

E.3 Risk Detection Implementation

1. Enable Identity Protection features (requires Premium P2):
    - Sign-in risk policies
    - User risk policies
    - MFA registration policy
2. Configure risk level thresholds:
    - High risk: Block access
    - Medium risk: Require MFA
    - Low risk: Allow with monitoring
3. Implement risk remediation workflows:
    - User notification process
    - Self-service remediation options
    - Administrative intervention procedures

STANDARD OPERATING PROCEDURES

A. SERVICE MANAGEMENT FRAMEWORK

A.1 Organizational Structure and Roles

The Azure B2C service requires a well-defined team structure with clear roles and responsibilities:

Role	Responsibilities	Department	Access Level
Service Owner	Strategic oversight, compliance accountability	IT Director	Read
Platform Administrator	Configuration management, updates, monitoring	Identity Team	Global Admin

Role	Responsibilities	Department	Access Level
Identity Operations	Day-to-day management, application onboarding	Identity Team	B2C Admin
Security Officer	Risk assessment, policy enforcement	Security Team	Read/Audit
Application Owner	Integration requirements, application-specific policies	Business Units	No direct access
Help Desk (Tier 1)	Basic troubleshooting, user assistance	IT Support	Directory Reader
Help Desk (Tier 2)	Advanced troubleshooting, application configuration	Identity Team	Application Admin

A.2 Service Level Objectives

The Azure B2C service will maintain the following SLOs:

Metric	Target	Measurement Method
Service Availability	99.9%	Azure Monitor uptime tracking
Authentication Success Rate	99.5%	Sign-in logs analysis
Response Time - Authentication	< 2 seconds	Application Insights
Response Time - Self-Service	< 5 seconds	Application Insights
Incident Response Time (Critical)	< 30 minutes	ITSM metrics
Incident Resolution Time (Critical)	< 4 hours	ITSM metrics
Change Implementation	< 5 business days	Change management system

A.3 Capacity Management

1. Monitor resource utilization:
   - Authentication transactions per day/month
   - Directory object count
   - API request volume
2. Establish thresholds for scaling:

- Alert at 70% of quota utilization
- Plan expansion at 80% of quota utilization

3. Quarterly capacity review process:
   - Analyze growth trends
   - Forecast future requirements
   - Adjust licensing and resources as needed

B. OPERATIONAL PROCEDURES

B.1 Application Onboarding Process

1. **Intake Phase**:
   - Application owner submits formal request via self-service portal
   - Required information: application details, integration type, user volumes
   - Security assessment performed for risk classification
   - Architecture review to validate integration pattern
2. **Implementation Phase**:
   - Create App Registration following naming standards
   - Configure appropriate authentication flow
   - Implement required custom claims
   - Store credentials in Key Vault
   - Document configuration in CMDB
3. **Validation Phase**:
   - Test authentication flows in non-production environment
   - Validate security controls and compliance
   - Perform user acceptance testing
   - Obtain security approval for production deployment
4. **Production Deployment**:
   - Implement via change management process
   - Provide integration details to application team
   - Monitor initial authentication activity
   - Conduct post-implementation review

B.2 Credential Management

1. **Secret Rotation Schedule**:
   - Client secrets: Every 180 days
   - Certificates: Before expiration (minimum 30 days)
   - TLS certificates: Annual renewal

2. **Rotation Process**:
    - Create new credential in B2C
    - Update reference in Key Vault
    - Configure application to use new credential
    - Verify functionality with new credential
    - Remove old credential after transition period
3. **Emergency Rotation**:
    - Defined process for compromised credentials
    - Immediate revocation capabilities
    - Notification procedures for affected applications

B.3 User Account Management

1. **External User Management**:
    - Self-service registration via user flows
    - Self-service password reset
    - Account closure process
    - Dormant account deactivation (365 days)
2. **Administrative Account Management**:
    - Quarterly access review for all privileged accounts
    - Just-in-time access for administrative operations
    - Separation of duties enforcement
    - Comprehensive audit logging
3. **Service Account Management**:
    - Dedicated service principals for automated processes
    - Least privilege access model
    - Quarterly review and recertification
    - Automated monitoring for suspicious activity

B.4 Incident Response

1. **Detection Capabilities**:
    - Automated alerts for security events
    - User-reported issues triage
    - Regular security reviews
2. **Response Procedures**:
    - Severity classification matrix
    - Defined escalation paths
    - Communication templates

- Containment strategies

3. **Recovery Procedures**:
    - Service restoration priorities
    - User communication process
    - Post-incident review
    - Lessons learned documentation

C. SELF-SERVICE PORTAL FOR APPLICATION TEAMS

The department will implement a self-service portal for application teams to request and manage their Azure B2C integration:

C.1 Portal Capabilities

1. **Application Registration Requests**:
    - Standardized form for new application onboarding
    - Integration type selection (web, SPA, API)
    - Required claims specification
    - Environment selection (Dev/Test/Prod)
2. **Integration Management**:
    - View existing application integrations
    - Request credential rotation
    - Update redirect URIs
    - Monitor application usage statistics
3. **Documentation and Guidance**:
    - Integration patterns and code samples
    - Troubleshooting guides
    - Best practices documentation
    - Security requirements reference
4. **Support Request Management**:
    - Ticketing interface for B2C-related issues
    - Status tracking for open requests
    - Knowledge base for common issues
    - Escalation path for complex problems

C.2 Portal Implementation

1. **Technology Platform**:
    - Power Pages portal with Azure AD authentication

    - Integration with departmental ITSM system
    - Automated workflows using Power Automate
    - Backend data in Dataverse
2. **Security Controls**:
    - Role-based access control
    - Audit logging for all portal actions
    - Approval workflows for sensitive operations
    - Data encryption for all stored information
3. **Integration with Azure B2C**:
    - Automated provisioning via Microsoft Graph API
    - Read-only dashboard for application metrics
    - Alert notification capabilities
    - Change request implementation tracking

D. CHANGE MANAGEMENT

D.1 Change Types and Classification

Change Type	Description	Approval Path	Implementation Window
Standard	Routine changes with established procedures	Team Lead	Regular maintenance window
Normal	Significant changes requiring planning	Change Advisory Board	Scheduled maintenance window
Emergency	Urgent changes to restore service	Emergency CAB	As needed with post-implementation review

D.2 Change Implementation Process

1. **Request and Assessment**:
    - Formal change request submission
    - Impact analysis and risk assessment
    - Documentation of implementation plan
    - Rollback procedure definition
2. **Approval Process**:
    - Technical review by identity team
    - Security review for compliance impact

   - Business approval for user-facing changes
   - Final authorization by appropriate authority
3. **Implementation**:
   - Pre-implementation testing in non-production
   - Scheduled deployment in maintenance window
   - Verification testing post-implementation
   - Documentation update in CMDB
4. **Review and Closure**:
   - Confirmation of successful implementation
   - Incident review for any issues encountered
   - Lessons learned documentation
   - Procedure updates if required

E. CONTINUOUS IMPROVEMENT

E.1 Key Performance Indicators

1. **Operational Metrics**:
   - Authentication success rate
   - Average response time
   - Error rates by application
   - Self-service utilization
2. **Security Metrics**:
   - Risk events detected
   - MFA adoption rate
   - Policy compliance percentage
   - Credential rotation compliance
3. **User Experience Metrics**:
   - Authentication completion rate
   - Average authentication time
   - Password reset success rate
   - User satisfaction surveys

E.2 Review Cycle

1. **Monthly Operational Review**:
   - Performance metrics analysis
   - Incident review and trending
   - Capacity planning updates

    - Short-term improvement actions
2. **Quarterly Security Review**:
    - Risk assessment update
    - Compliance verification
    - Threat intelligence review
    - Security control effectiveness
3. **Annual Service Review**:
    - Strategic alignment assessment
    - Technology roadmap update
    - Major enhancement planning
    - Long-term improvement strategy

APPENDICES

APPENDIX A: REFERENCE ARCHITECTURE DIAGRAMS

A.1 Logical Architecture

[Detailed logical architecture diagram showing all components and their relationships]

A.2 Network Flow Diagrams

[Network flow diagrams for each integration pattern]

A.3 Security Architecture

[Security architecture diagram highlighting control points and data protection]

APPENDIX B: SAMPLE CODE FOR INTEGRATIONS

B.1 .NET Core Integration

```
// ASP.NET Core Web App with Azure B2C Authentication
// File: Program.cs

using Microsoft.AspNetCore.Authentication.OpenIdConnect;
using Microsoft.Identity.Web;
using Microsoft.Identity.Web.UI;

var builder = WebApplication.CreateBuilder(args);

// Add Azure B2C authentication
builder.Services.AddAuthentication(OpenIdConnectDefaults.AuthenticationScheme)
    .AddMicrosoftIdentityWebApp(options =>
    {
        builder.Configuration.Bind("AzureAdB2C", options);

        options.Events = new OpenIdConnectEvents
        {
            OnSignedOutCallbackRedirect = (context) =>
            {
                context.Response.Redirect("/");
                context.HandleResponse();
                return Task.CompletedTask;
            }
        };
    });

// Add MVC controllers and views
builder.Services.AddControllersWithViews()
    .AddMicrosoftIdentityUI();

// Add authorization policies
builder.Services.AddAuthorization(options =>
{
    // Default policy requiring authentication
    options.FallbackPolicy = options.DefaultPolicy;
});

// Add session state
builder.Services.AddRazorPages();
builder.Services.AddSession(options =>
{
    options.IdleTimeout = TimeSpan.FromMinutes(60);
    options.Cookie.HttpOnly = true;
    options.Cookie.IsEssential = true;
    options.Cookie.SecurePolicy = CookieSecurePolicy.Always;
    options.Cookie.SameSite = SameSiteMode.None;
});

var app = builder.Build();

// Configure the HTTP request pipeline
if (!app.Environment.IsDevelopment())
{
    app.UseExceptionHandler("/Error");
    app.UseHsts();
}

app.UseHttpsRedirection();
app.UseStaticFiles();
app.UseRouting();

app.UseAuthentication();
app.UseAuthorization();

app.UseSession();

app.MapRazorPages();
app.MapControllers();

app.Run();
```

```
// File: appsettings.json

{
  "AzureAdB2C": {
    "Instance": "https://auth.{DEPARTMENT}.canada.ca/",
    "TenantId": "{TENANT_ID}",
    "ClientId": "{CLIENT_ID}",
    "ClientSecret": "{CLIENT_SECRET}",
    "CallbackPath": "/signin-oidc",
    "SignedOutCallbackPath": "/signout-callback-oidc",
    "SignUpSignInPolicyId": "{POLICY_NAME}",
    "ResetPasswordPolicyId": "{RESET_POLICY_NAME}",
    "EditProfilePolicyId": "{EDIT_PROFILE_POLICY_NAME}"
  },
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*"
}
```

```
// File: AccountController.cs

using Microsoft.AspNetCore.Authentication;
using Microsoft.AspNetCore.Authentication.OpenIdConnect;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Identity.Web;

[AllowAnonymous]
public class AccountController : Controller
{
    private readonly IConfiguration _configuration;

    public AccountController(IConfiguration configuration)
    {
        _configuration = configuration;
    }

    [HttpGet]
    public IActionResult SignIn()
    {
        var redirectUrl = Url.Action("Index", "Home");
        return Challenge(
            new AuthenticationProperties { RedirectUri = redirectUrl },
            OpenIdConnectDefaults.AuthenticationScheme);
    }

    [HttpGet]
    public IActionResult ResetPassword()
    {
        var redirectUrl = Url.Action("Index", "Home");
        var properties = new AuthenticationProperties { RedirectUri = redirectUrl };
        properties.Items[Constants.Policy] = _configuration["AzureAdB2C:ResetPasswordPoli
        return Challenge(properties, OpenIdConnectDefaults.AuthenticationScheme);
    }

    [HttpGet]
    public IActionResult EditProfile()
    {
        var redirectUrl = Url.Action("Index", "Home");
        var properties = new AuthenticationProperties { RedirectUri = redirectUrl };
        properties.Items[Constants.Policy] = _configuration["AzureAdB2C:EditProfilePolicy
        return Challenge(properties, OpenIdConnectDefaults.AuthenticationScheme);
        }

        [HttpGet]
        public IActionResult SignOut()
        {
            var callbackUrl = Url.Action("SignedOut", "Account", values: null, protocol: Requ
            return SignOut(
                new AuthenticationProperties { RedirectUri = callbackUrl },
                OpenIdConnectDefaults.AuthenticationScheme,
                CookieAuthenticationDefaults.AuthenticationScheme);
        }

        [HttpGet]
        public IActionResult SignedOut()
        {
            if (User.Identity.IsAuthenticated)
            {
                // Redirect to home page if user is authenticated
                return RedirectToAction("Index", "Home");
            }

            return View();
        }

        [HttpGet]
        [Authorize]
        public IActionResult Profile()
        {
            return View(User.Claims);
        }
    }
```

B.2 Node.js Integration

```javascript
// Node.js Express App with Azure B2C Authentication
// File: app.js

const express = require('express');
const session = require('express-session');
const passport = require('passport');
const BearerStrategy = require('passport-azure-ad').BearerStrategy;
const OIDCStrategy = require('passport-azure-ad').OIDCStrategy;
const dotenv = require('dotenv');
const path = require('path');

// Load environment variables
dotenv.config();

// Create Express app
const app = express();

// Configure session middleware
app.use(session({
  secret: process.env.SESSION_SECRET,
  resave: false,
  saveUninitialized: false,
  cookie: {
    secure: true,
    httpOnly: true,
    maxAge: 3600000 // 1 hour
  }
}));

// Configure view engine
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');

// Configure Passport middleware
app.use(passport.initialize());
app.use(passport.session());

// Configure passport OIDC strategy for web app authentication
passport.use('oidc', new OIDCStrategy({
  identityMetadata: `https://auth.${process.env.DEPARTMENT}.canada.ca/${process.env.TENAN
  clientID: process.env.CLIENT_ID,
  clientSecret: process.env.CLIENT_SECRET,
  responseType: 'code',
  responseMode: 'form_post',
  redirectUrl: process.env.REDIRECT_URI,
  allowHttpForRedirectUrl: false,
  validateIssuer: true,
  isB2C: true,
  passReqToCallback: false,
  scope: ['openid', 'profile'],
  loggingLevel: 'info'
}, (profile, done) => {
  // Passport callback after authentication
  return done(null, profile);
}));

// Configure passport Bearer strategy for API authentication
passport.use('bearer', new BearerStrategy({
  identityMetadata: `https://auth.${process.env.DEPARTMENT}.canada.ca/${process.env.TENAN
  clientID: process.env.CLIENT_ID,
  validateIssuer: true,
  issuer: `https://auth.${process.env.DEPARTMENT}.canada.ca/${process.env.TENANT}/v2.0/`,
  passReqToCallback: false,
  loggingLevel: 'info'
}, (token, done) => {
  // Passport callback for API calls
  return done(null, token, token);
}));

// Serialize and deserialize user
passport.serializeUser((user, done) => {
  done(null, user);
});

passport.deserializeUser((user, done) => {
  done(null, user);
});

// Authentication middleware
const ensureAuthenticated = (req, res, next) => {
  if (req.isAuthenticated()) {
    return next();
  }
  res.redirect('/login');
```

```
};

// Routes
app.get('/', (req, res) => {
  res.render('index', { user: req.user });
});

app.get('/login', passport.authenticate('oidc', {
  successRedirect: '/',
  failureRedirect: '/login-failed'
}));

app.post('/auth/openid/return', passport.authenticate('oidc', {
  successRedirect: '/',
  failureRedirect: '/login-failed'
}));

app.get('/profile', ensureAuthenticated, (req, res)
```

APPENDIX D: REFERENCE DOCUMENTS

D.1 Government of Canada Resources

- ITSG-33: IT Security Risk Management
- Cloud Security Control Profile for Protected B Information
- GC Digital Standards
- Official Languages Requirements for External Services

D.2 Microsoft Documentation

- Azure AD B2C Technical Documentation
- Azure Architecture Center Reference Architectures
- Microsoft Identity Platform Best Practices

D.3 Industry Standards

- ISO/IEC 27001:2013 Control Mappings
- NIST SP 800-63 Digital Identity Guidelines
- OWASP Authentication Security Best Practices

APPENDIX E: GLOSSARY OF TERMS

Term	Definition
AAD	Azure Active Directory, Microsoft's cloud identity service
AFD	Azure Front Door, a global content delivery network service
App Registration	A record in Azure AD representing an application that can use Azure AD for authentication
Azure B2C	Azure Active Directory Business-to-Consumer, a customer identity access management solution
Azure App Gateway	Application delivery controller service providing layer 7 load balancing, WAF, and SSL termination
CATS	Canadian Authentication and Trust Services, a standard for government authentication
CDN	Content Delivery Network, a distributed server network that delivers web content to users
Claims	Pieces of information about a user that are passed between identity provider and application
Conditional Access	A feature of Azure AD that controls access to applications based on specific conditions
CSR	Certificate Signing Request, a request for a digital certificate from a certificate authority
Custom Domain	A domain name configured for Azure B2C instead of the default *.b2clogin.com domain
Custom Policy	XML files that define the behavior of Azure B2C authentication experiences
EAB	Enterprise Access Broker, SSC's OIDC broker service for GCKey and Sign-in Partners
Front Channel Logout	Browser-based logout mechanism that notifies all applications of a user's logout event

Term	Definition
GCKey	Government of Canada credential service providing secure access to online government services
GCCF	Government of Canada Credential Federation, the federation of GC credential services
Identity Experience Framework (IEF)	The framework used by Azure B2C for customizing authentication experiences
Identity Provider (IdP)	A system that creates, maintains and manages identity information and authentication services
IDP	Identity Provider, such as GCKey, Sign-in Partner, or other social identity providers
ITSG-33	IT Security Guidance document by CSE defining security control profiles for the GC
JWT	JSON Web Token, a compact, URL-safe means of representing claims to be transferred between parties
Key Vault	Azure service for securely storing and accessing secrets like certificates and API keys
MFA	Multi-Factor Authentication, requiring two or more verification methods for authentication
OAuth 2.0	Authorization framework that enables applications to obtain limited access to user accounts
OIDC	OpenID Connect, an authentication layer built on top of OAuth 2.0
OpenID Connect	Identity layer on top of OAuth 2.0 that allows clients to verify user identity
PAI	Persistent Anonymous Identifier, a unique identifier for users that doesn't reveal personal information
Protected B	Government of Canada information classification requiring medium security controls
Relying Party (RP)	Application that relies on an identity provider for authentication

Term	Definition
SAML 2.0	Security Assertion Markup Language, an XML-based protocol for authentication and authorization
SIC	Sign-in Canada, TBS's OIDC broker service for GCKey and Sign-in Partners
Sign-in Partners	Canadian financial institutions that provide credential services for government authentication
SSO	Single Sign-On, authentication process allowing users to access multiple applications with one login
TLS	Transport Layer Security, cryptographic protocol for secure communications over a network
User Flow	Pre-defined authentication experience path in Azure B2C
WAF	Web Application Firewall, a security solution that monitors and filters HTTP traffic

APPENDIX F: REFERENCE DOCUMENTS

Government of Canada Resources

- ITSG-33: IT Security Risk Management
- ITSG-22: Baseline Security Requirements for Network Security Zones
- Cloud Security Control Profile for Protected B Information
- GC Digital Standards
- Official Languages Requirements for External Services
- Government of Canada Web Standards
- Directive on Service and Digital
- Standard on Web Accessibility
- Standard on Optimizing Websites and Applications for Mobile Devices
- Guidance on Cloud Authentication for the Government of Canada

Microsoft Azure B2C Documentation

Core Documentation

- Azure AD B2C Documentation
- What is Azure AD B2C?
- Azure AD B2C Pricing
- Azure AD B2C Service Limits and Restrictions

Architecture and Implementation

- Azure AD B2C Architecture Overview
- Creating an Azure AD B2C Tenant
- Custom Domain Configuration
- Azure Front Door with B2C
- Azure Application Gateway Documentation

User Authentication and Policies

- Azure AD B2C User Flows
- Azure AD B2C Custom Policies
- Identity Experience Framework
- Adding External Identity Providers
- User Migration to Azure AD B2C
- SAML Service Provider Configuration

Security and Compliance

- Azure AD B2C MFA Configuration
- Azure AD B2C Identity Protection
- Azure Key Vault Integration
- Microsoft Entra Permissions Management
- Conditional Access for B2C
- Restricting Azure AD Access by Location

Application Integration

- B2C App Integration Patterns
- Register a Web Application

- Single-Page Application Integration
- Mobile and Desktop App Integration
- API Authorization with Azure AD B2C
- Power Pages Portal Integration

Monitoring and Operations

- B2C Audit Logs and Monitoring
- User Behavior Analytics
- Azure Monitor for B2C
- B2C Operational Readiness Checklist
- Troubleshooting Azure AD B2C

Token and Protocol Reference

- B2C Token Reference
- OpenID Connect Protocol
- OAuth 2.0 Authorization Code Flow
- JWT Validation

Industry Standards and Specifications

- ISO/IEC 27001:2013 Information Security Management
- ISO/IEC 27017:2015 Cloud Security
- ISO/IEC 27018:2019 Protection of PII in Public Clouds
- NIST SP 800-63 Digital Identity Guidelines
- OpenID Connect Specifications
- OAuth 2.0 Specifications
- FIDO2 Authentication Standards
- OWASP Authentication Security Best Practices
- OWASP SAML Security Cheat Sheet

Code Samples and SDKs

- Microsoft Identity Platform Code Samples
- Microsoft Authentication Library (MSAL)
- Azure AD B2C .NET Code Samples
- Azure AD B2C JavaScript/SPA Code Samples

- Azure AD B2C Mobile Code Samples
- Azure AD B2C Custom Policy Samples

Whitepapers and Advanced Topics

- Azure AD B2C FAQ
- Azure Security Benchmarks
- Azure Identity Management Best Practices
- Zero Trust Identity and Access Management
- Microsoft Security Documentation