# XFUEL PLATFORM

## SSO Integration Guide

For Prometheus HUB Partners & 3rd Party Integrations

Version 1.0 • April 2026
Prometheus Pro, LLC • Confidential

# Table of Contents

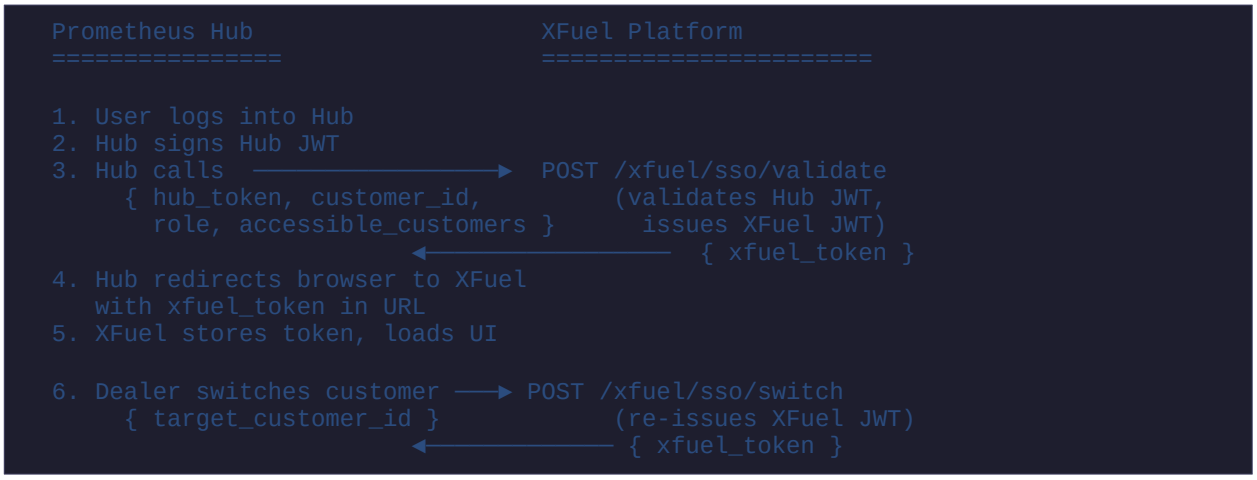
# 1 Overview & Architecture

XFuel is a fleet fuel-card management platform hosted at xfuel.prometheuspro.us. It uses JSON Web Tokens (JWT) for authentication. Prometheus Hub acts as the Identity Provider (IdP); XFuel trusts Hub-issued tokens and exchanges them for XFuel-scoped tokens.

### Key Concepts

| Term | Definition |
|---|---|
| **Hub Token** | JWT issued by Prometheus Hub, signed with PORTAL_SECRET_KEY. Short-lived (1 h). Proves the user is authenticated in Hub. |
| **XFuel Token** | JWT issued by XFuel after validating the Hub token. Contains customer_id, role, and accessible_customers. Valid for 8 hours. |
| **customer_id** | Unique identifier for a fleet customer (e.g., "tmodal"). Maps to a row in xfuel_users. |
| **role** | Either "user" (standard access), "sub_dealer" (can view multiple customers), or "dealer" (full multi-account management). |
| **SSO Switch** | A dealer/sub_dealer re-issuing their own XFuel token to act as a different customer without returning to Hub. |

### Architecture Diagram



# 2 Authentication Flow

The SSO flow is a three-step process: Hub authenticates the user, Hub validates the token with XFuel, and XFuel issues its own token.

### Step-by-step

### Step 1 – Hub Authentication

User signs in to Prometheus Hub using their standard Hub credentials. Hub issues a short-lived Hub JWT (1 hour) signed with PORTAL_SECRET_KEY.

### Step 2 – Token Exchange

Hub's backend calls POST /xfuel/sso/validate with the Hub JWT and the customer_id for which access is being granted. XFuel verifies the Hub JWT signature, checks that customer_id exists, then issues an XFuel JWT (8 hours).

### Step 3 – Redirect

Hub redirects the user's browser to https://xfuel.prometheuspro.us with the xfuel_token in the URL fragment or as a query parameter. XFuel's login.html stores the token in localStorage and loads the dashboard.

### Step 4 – Token Use

All subsequent API requests from the browser include the XFuel JWT in the Authorization header: Bearer <xfuel_token>. The token encodes customer_id, role, and accessible_customers — no session state is stored on the server.

### Step 5 – Refresh

Before the 8-hour window expires, the client may call POST /xfuel/sso/refresh with the current XFuel token to receive a fresh token. Hub does not need to be involved.

> ⚠ **Note:** Hub must call /xfuel/sso/validate from its backend — never from the browser — to keep the hub_token server-side.

# 3 API Endpoints Reference

**i Info:** Base URL for all endpoints: https://xfuel.prometheuspro.us/api All endpoints require Content-Type: application/json. Authenticated endpoints require: Authorization: Bearer <xfuel_token>

## 3.1 POST /xfuel/sso/validate

**POST** `/xfuel/sso/validate`

Validates a Prometheus Hub JWT and issues an XFuel JWT. Call this from Hub's backend immediately after a user completes Hub login. This is the primary SSO entry point.

### Request Body

| Field | Type | Description |
|---|---|---|
| `hub_token` | string (req) | The JWT issued by Prometheus Hub, signed with PORTAL_SECRET_KEY. |
| `customer_id` | string (req) | The XFuel customer account to log into (must exist in xfuel_users). |
| `role` | string | "user" (default), "sub_dealer", or "dealer". Controls access level. |
| `carrier_name` | string | Human-readable company name displayed in the XFuel sidebar. |
| `redirect_url` | string | Optional. Returned as-is for Hub to redirect the browser after token exchange. |
| `accessible_customers` | array | Required for dealer/sub_dealer. List of {id, name, role} objects the user can switch to. |

### Response Fields

| Field | Type | Description |
|---|---|---|
| `xfuel_token` | string | Signed XFuel JWT. Valid for 8 hours. |
| `customer_id` | string | Echo of the customer_id used. |
| `expires_in` | integer | 28800 (seconds = 8 hours). |
| `role` | string | Role embedded in the token. |
| `accessible_customers` | array | Echo of the accessible_customers passed in. Embedded in token. |

### Example Request

```
POST https://xfuel.prometheuspro.us/api/xfuel/sso/validate
Content-Type: application/json
```

```
{
  "hub_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "customer_id": "tmodal",
  "role": "dealer",
  "carrier_name": "T Modal Trucking",
  "accessible_customers": [
    { "id": "tmodal",   "name": "T Modal Trucking",    "role": "dealer" },
    { "id": "palmetto", "name": "Palmetto Transport",  "role": "user"   },
    { "id": "sunstate", "name": "Sun State Logistics", "role": "user"   }
  ]
}
```

### Example Response

```
{
  "xfuel_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "customer_id": "tmodal",
  "expires_in": 28800,
  "role": "dealer",
  "accessible_customers": [
    { "id": "tmodal",   "name": "T Modal Trucking",    "role": "dealer" },
    { "id": "palmetto", "name": "Palmetto Transport",  "role": "user"   },
    { "id": "sunstate", "name": "Sun State Logistics", "role": "user"   }
  ]
}
```

**i Info:** For plain "user" accounts, omit accessible_customers entirely (or pass an empty array).

## 3.2 POST /xfuel/sso/refresh

**POST** `/xfuel/sso/refresh`

Renews an existing XFuel JWT without requiring Hub re-authentication. Use this to silently extend sessions before the 8-hour window expires.

### Request Body

| Field | Type | Description |
|---|---|---|
| `token` | string (req) | The current (still-valid) XFuel JWT. |

### Response Fields

| Field | Type | Description |
|---|---|---|
| `xfuel_token` | string | New XFuel JWT with a fresh 8-hour expiry. |
| `expires_in` | integer | 28800. |

### Example Request

```
POST https://xfuel.prometheuspro.us/api/xfuel/sso/refresh
Content-Type: application/json

{
  "token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."
}
```

### Example Response

```
{
  "xfuel_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "expires_in": 28800
}
```

ℹ **Info:** Refresh will fail (401) if the existing token is already expired. Hub should schedule refresh at ~7 hours.

## 3.3 POST /xfuel/sso/switch

**POST** `/xfuel/sso/switch`

Allows a dealer or sub_dealer to switch into a different customer account without returning to Prometheus Hub. XFuel validates that the target customer is in the caller's accessible_customers list and issues a new XFuel JWT scoped to that account.

### Request Body

| Field | Type | Description |
|---|---|---|
| `target_customer_id` | string (req) | The customer_id to switch into. Must be in the token's accessible_customers list. |

### Response Fields

| Field | Type | Description |
|---|---|---|
| `xfuel_token` | string | New XFuel JWT scoped to target_customer_id. |
| `customer_id` | string | Echo of target_customer_id. |
| `carrier_name` | string | Human-readable name for the switched-to account. |
| `expires_in` | integer | 28800. |

### Example Request

```
POST https://xfuel.prometheuspro.us/api/xfuel/sso/switch
Content-Type: application/json
Authorization: Bearer <current_xfuel_token>

{
  "target_customer_id": "palmetto"
}
```

### Example Response

```
{
  "xfuel_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "customer_id": "palmetto",
  "carrier_name": "Palmetto Transport",
  "expires_in": 28800
}
```

**Info:** Returns 403 if the target is not in accessible_customers. The new token inherits the full accessible_customers list so the dealer can switch again.

## 3.4 POST /xfuel/sso/logout

**POST** `/xfuel/sso/logout`

Invalidates the current XFuel session. Because XFuel tokens are stateless JWTs, logout primarily clears the client-side token. The server records the logout event.

### Request Body

| Field | Type | Description |
|---|---|---|
| — (none) | — | No request body required. Token is read from Authorization header. |

### Response Fields

| Field | Type | Description |
|---|---|---|
| message | string | "Logged out successfully" |

### Example Request

```
POST https://xfuel.prometheuspro.us/api/xfuel/sso/logout
Authorization: Bearer <xfuel_token>
```

### Example Response

```
{
  "message": "Logged out successfully"
}
```

**Info:** After calling logout, Hub should also clear its own session and redirect to the Hub login page.

# 4 Role Hierarchy & Permissions

XFuel supports three roles in a parent-child hierarchy. The role is embedded in the XFuel JWT and controls which UI components and API actions are available.

### Role Comparison

| Feature | user | sub_dealer | dealer |
|---|---|---|---|
| View own fuel cards | ✓ | ✓ | ✓ |
| View fleet vehicles | ✓ | ✓ | ✓ |
| Customer switcher UI | – | ✓ | ✓ |
| Switch to sub-accounts | – | Own list | Full hierarchy |
| Add users | – | – | ✓ |
| accessible_customers | Empty | Subset | Full list |
| parent_customer_id | null | dealer id | self |

### Hierarchy Example

```
T Modal Trucking  (role: dealer)
├── T Modal HQ        (customer_id: "tmodal",   role: dealer)
├── Palmetto Transport  (customer_id: "palmetto", role: user)
└── Sun State Logistics (customer_id: "sunstate", role: user)

When T Modal logs in with role: dealer, accessible_customers includes
all three accounts. The customer switcher dropdown appears in the sidebar.
```

# 5 Step-by-Step Integration Guide

## Step 1: Create Customer Account in XFuel

Before SSO can work, the customer_id must exist in the xfuel_users table.

Contact XFuel support or use the admin panel to create the account. Required fields:

```
-- Example (run on XFuel database server)
INSERT INTO xfuel_users (email, hashed_password, customer_id, carrier_name,
is_active)
VALUES ('admin@yourcompany.com', '<bcrypt_hash>', 'yourcompany', 'Your Company
Name', true);
```

**Info:** The email is used as the login fallback. The hashed_password is only needed if direct (non-SSO) login is also required.

## Step 2: Configure Hub with XFuel SSO Credentials

Hub needs two configuration values to talk to XFuel:

```
XFUEL_SSO_URL=https://xfuel.prometheuspro.us/api/xfuel/sso/validate
XFUEL_REDIRECT_URL=https://xfuel.prometheuspro.us/login.html

# Hub signs its tokens with PORTAL_SECRET_KEY.
# XFuel uses the same value as PROMETHEUS_HUB_SECRET.
# Make sure both match — mismatched secrets cause 401 errors.
```

## Step 3: Implement the Token Exchange (Hub Backend)

When a Hub user clicks "Open XFuel" or navigates to the fuel management section, Hub's backend should:

```
# Python example (Hub backend)
import httpx, jwt, os

async def open_xfuel(hub_user, customer_id, accessible_customers):
    hub_token = jwt.encode(
        {"sub": hub_user.id, "email": hub_user.email, "exp": time.time()+3600},
        os.environ["PORTAL_SECRET_KEY"], algorithm="HS256"
    )
    resp = await httpx.post(
        "https://xfuel.prometheuspro.us/api/xfuel/sso/validate",
        json={
            "hub_token": hub_token,
            "customer_id": customer_id,
            "role": hub_user.xfuel_role,
            "carrier_name": hub_user.company_name,
            "accessible_customers": accessible_customers,
        }
    )
    xfuel_token = resp.json()["xfuel_token"]
    redirect = f"https://xfuel.prometheuspro.us/login.html?token={xfuel_token}"
    return RedirectResponse(redirect)
```

**Note:** Never expose hub_token to the browser. The call to /xfuel/sso/validate must happen server-side.

## Step 4: Pass Token to XFuel Frontend

XFuel's login.html accepts the token in the URL:

```
# Redirect URL pattern
https://xfuel.prometheuspro.us/login.html?token=<xfuel_token>

# login.html automatically:
```

```
#   1. Reads ?token= from the URL
#   2. Stores it in localStorage as "xfuel_token"
#   3. Redirects to dashboard.html
#   4. Clears the token from the URL bar
```

### Step 5: Handle Token Refresh (Optional)

XFuel tokens last 8 hours. For long sessions, implement refresh:

```
# Check expiry on each Hub page load
payload = jwt.decode(xfuel_token, options={"verify_signature": False})
if payload["exp"] - time.time() < 3600:  # less than 1 hour left
    resp = httpx.post(
        "https://xfuel.prometheuspro.us/api/xfuel/sso/refresh",
        json={"token": xfuel_token}
    )
    xfuel_token = resp.json()["xfuel_token"]
    # Update stored token
```

### Step 6: Implement Logout

When the user logs out of Hub, also log them out of XFuel:

```
# Call XFuel logout endpoint
httpx.post(
    "https://xfuel.prometheuspro.us/api/xfuel/sso/logout",
    headers={"Authorization": f"Bearer {xfuel_token}"}
)

# In the browser, also clear localStorage:
localStorage.removeItem("xfuel_token")
```

# 6 Testing with cURL

Use these commands to manually test each SSO endpoint from a terminal.

## 6.1 Validate (get XFuel token)

```
# 1. Generate a Hub token (requires PORTAL_SECRET_KEY)
HUB_TOKEN=$(python3 -c "
import jwt, time
print(jwt.encode({
  "sub": "test_user", "email": "test@prometheuspro.us",
  "exp": int(time.time())+3600
}, "YOUR_PORTAL_SECRET_KEY", algorithm="HS256"))
")

# 2. Exchange for XFuel token
curl -s -X POST https://xfuel.prometheuspro.us/api/xfuel/sso/validate \
  -H 'Content-Type: application/json' \
  -d '{
    "hub_token": "'$HUB_TOKEN'",
    "customer_id": "tmodal",
    "role": "user",
    "carrier_name": "T Modal Trucking"
  }' | python3 -m json.tool
```

## 6.2 Refresh

```
curl -s -X POST https://xfuel.prometheuspro.us/api/xfuel/sso/refresh \
  -H 'Content-Type: application/json' \
  -d '{"token": "'$XFUEL_TOKEN'"}' | python3 -m json.tool
```

## 6.3 Switch Customer (dealer only)

```
# XFUEL_TOKEN must be a dealer/sub_dealer token
curl -s -X POST https://xfuel.prometheuspro.us/api/xfuel/sso/switch \
  -H 'Content-Type: application/json' \
  -H "Authorization: Bearer $XFUEL_TOKEN" \
  -d '{"target_customer_id": "palmetto"}' | python3 -m json.tool
```

## 6.4 Logout

```
curl -s -X POST https://xfuel.prometheuspro.us/api/xfuel/sso/logout \
  -H "Authorization: Bearer $XFUEL_TOKEN"
```

## 6.5 Verify Token Payload

```
# Decode XFuel token without verification (inspect claims)
python3 -c "
import jwt, sys
payload = jwt.decode(sys.argv[1], options={'verify_signature': False})
import json; print(json.dumps(payload, indent=2))
" $XFUEL_TOKEN
```

# 7 Troubleshooting

| Error / Symptom | Likely Cause | Resolution |
|---|---|---|
| 401 Unauthorized on /sso/validate | Hub token signature invalid | Confirm PORTAL_SECRET_KEY on Hub matches PROMETHEUS_HUB_SECRET on XFuel server. Check /opt/prometheus-ai/.env |
| 401 — "Token expired" | Hub token older than 1 hour | Generate a fresh Hub token. Tokens are valid for 3600 seconds. |
| 422 Unprocessable Entity | Missing required field or wrong JSON type | Check that hub_token and customer_id are included. accessible_customers must be an array of objects, not strings. |
| 403 on /sso/switch | target_customer_id not in accessible_customers | Ensure the original /sso/validate call included the target in accessible_customers. |
| Dashboard shows blank after SSO redirect | xfuel_token missing or malformed in URL | Check the redirect URL format. Token must be in ?token= query param. Open browser DevTools > Application > Local Storage to confirm "xfuel_token" was set. |
| Customer switcher not visible | Role is "user" or token missing accessible_customers | Re-validate with role: "dealer" and a non-empty accessible_customers array. |
| API calls return 401 after 8 hours | XFuel token expired | Implement token refresh (Section 3.2) or re-run the SSO validate flow. |
| Vehicles page shows blank for new customer | No vehicles enrolled for that customer_id | Contact XFuel support to enroll vehicles under the customer account. |

# 8 Security Best Practices

### Never send Hub tokens to the browser

PORTAL_SECRET_KEY is a server-side secret. The Hub token must only travel from Hub's backend to XFuel's backend. If it is exposed to the browser it can be replayed.

### Use HTTPS everywhere

All API calls must use https://. HTTP calls will be rejected by the XFuel nginx configuration.

### Rotate secrets periodically

PORTAL_SECRET_KEY / PROMETHEUS_HUB_SECRET should be rotated at least annually. Coordinate the rotation between Hub and XFuel ops teams to avoid a service interruption.

### Validate accessible_customers server-side

XFuel enforces that /sso/switch targets are within accessible_customers, but Hub should also validate before calling validate. Never trust the browser to supply the accessible_customers list.

### Log SSO events

Hub and XFuel both log SSO validate and switch events. If a customer reports unexpected access, review the XFuel API access log at /var/log/nginx/access.log on the server.

### Short Hub token lifetime

Keep Hub tokens short-lived (1 hour or less). XFuel tokens are intentionally longer (8 hours) to reduce re-authentication friction for drivers, but Hub tokens should be tight.

### Restrict /sso/validate by IP (optional)

For maximum security, configure the XFuel nginx to only accept calls to /xfuel/sso/validate from Hub's static egress IP. Contact XFuel ops to add this rule.

# Support Contacts

| | |
|---|---|
| **XFuel Support Email** | `support@prometheuspro.us` |
| **Server (SSH)** | `159.203.162.33 (internal access only)` |
| **API Base URL** | `https://xfuel.prometheuspro.us/api` |
| **API Docs** | `https://xfuel.prometheuspro.us/api-docs.html` |
| **Swagger JSON** | `https://xfuel.prometheuspro.us/xfuel-openapi.json` |
| **Test Account** | `carlo@tmodal.com / tmodal123  (customer_id: tmodal)` |