📄Documentation: AzureChatOpenAI and Bearer Token Authentication

Overview

This document clarifies that the `AzureChatOpenAI` wrapper in **LangChain** does **not** support **Bearer Token-based authentication** (e.g., tokens issued by Okta or other identity providers). It only supports **API key–based authentication**, as per Azure OpenAI and LangChain integration guidelines.

🕯️ What is `AzureChatOpenAI`?

`AzureChatOpenAI` is a LangChain wrapper class used to interface with the **Azure-hosted version of OpenAI's models** (e.g., `gpt-4`, `gpt-3.5-turbo`) via Azure's OpenAI service.

It relies on the `openai` Python SDK under the hood, which traditionally supports authentication using **API keys**, not OAuth2 or Bearer tokens.

🔐Authentication Mechanisms Supported

🔗Supported

- `openai_api_key`: Required and used to authenticate requests to Azure's OpenAI endpoint.
- `azure_endpoint`, `openai_api_version`, `deployment_name`: Required config parameters for Azure OpenAI.

🔩Not Supported

- `Authorization: Bearer <token>` headers (e.g., Okta access tokens)
- OAuth2 flows
- Injecting custom headers via LangChain APIs

📖Supporting Evidence

1. LangChain Source Code

AzureChatOpenAI Constructor Signature:

Source: [GitHub - AzureChatOpenAI]

```
class AzureChatOpenAI(BaseAzureOpenAI):
    def __init__(
        self,
        deployment_name: str,
        model_name: str,
        openai_api_key: Optional[str] = None,
        ...
    ):
```

- Only `openai_api_key` is accepted for authentication.
- No support for `Authorization` headers or Bearer tokens.

---

2. Official LangChain Documentation

Source: LangChain - Azure OpenAI Integration

- Documents `openai_api_key`, `deployment_name`, `openai_api_version`, and `azure_endpoint`.
- No mention of Bearer token support.

---

3. OpenAI Python SDK v1

Source: OpenAI Python SDK GitHub

- Auth is via:

  ```
  client = OpenAI(api_key="sk-...")
  ```

- No Bearer Token, OAuth2, or header injection support by default.

🛠️Workarounds

1. Use Raw HTTP Calls

Manually call Azure OpenAI endpoints using `requests` with Bearer tokens:

```
import requests

headers = {
    "Authorization": f"Bearer {okta_token}",
```

```
    "Content-Type": "application/json"
}

url = f"{azure_endpoint}/openai/deployments/{deployment_name}/chat/completions?
api-version={api_version}"

data = {
    "messages": [{"role": "user", "content": "Hello!"}],
    "temperature": 0.7
}

response = requests.post(url, headers=headers, json=data)
```

2. Extend LangChain with a Custom Wrapper

Subclass `AzureChatOpenAI` and inject a custom `client` that uses Bearer token headers.

🔗Recommendation

If your authentication mechanism uses Bearer tokens (e.g., from Okta or Azure AD), you **must bypass LangChain's default wrappers** and:

- Use `requests` or `httpx` for API calls, **or**
- Subclass `AzureChatOpenAI` and inject a customized `AzureOpenAI` client manually.

🕔Conclusion

`AzureChatOpenAI` in LangChain does **not support Bearer token authentication** out-of-the-box. Teams using OAuth2/Bearer tokens should use raw HTTP calls or implement a custom client layer.