API Discovery for Red Team Target Configuration

When you're red-teaming a new application/model, here's a systematic approach to discover the API format.

Phase 1: Reconnaissance

1. Browser DevTools Inspection

- Open the application in browser with DevTools (F12)
- Go to **Network** tab, filter for "Fetch/XHR"
- Interact with the AI feature (send a test message)
- Look for API calls - you'll see:
  - **Endpoint URL:** `https://api.example.com/v1/chat`
  - **HTTP Method:** Usually POST
  - **Request Headers:** Authorization, Content-Type, etc.
  - **Request Payload:** Click the call → Payload tab
  - **Response:** Click Response tab

2. cURL Replication

- Right-click the API call → **"Copy as cURL"**
- Test in terminal to confirm it works
- Strip unnecessary headers (keep Authorization, Content-Type)

3. Documentation Search

- Check if they have public API docs (common for SaaS AI tools)
- Search: `[company name] API documentation`
- Look for authentication method (Bearer tokens, API keys, OAuth)

Phase 2: Format Analysis

4. Request Format Extraction

Typical patterns you'll see:

OpenAI-compatible (most common):

```json
{
  "messages": [{"role": "user", "content": "test"}],
  "model": "gpt-4",
  "max_tokens": 500
}
```

Anthropic format:

```json
{
  "messages": [{"role": "user", "content": "test"}],
  "model": "claude-3-opus-20240229",
  "max_tokens": 1024
}
```

Custom application format:

```json
{
  "query": "test",
  "session_id": "abc123",
  "parameters": {"temperature": 0.7}
}
```

5. Response Format Extraction

Look for where the AI's response text lives:

OpenAI-compatible:

```json
{
  "choices": [{
    "message": {"content": "RESPONSE TEXT HERE"}
  }]
}
```

Custom format examples:

```JSON
{
  "data": {"response": "TEXT"},
  "metadata": {...}
}
{
  "result": "TEXT",
  "status": "success"
}
```

Phase 3: Prisma AIRS Configuration

6. Map to Prisma AIRS Fields

In Prisma AIRS red team target configuration:

Field	Description
Endpoint URL	The full URL from DevTools
Request JSON	Replace user content with `{INPUT}` placeholder
Response JSON Path	Use JSONPath to the response text field

Example mapping:

Request Template:

```JSON
{
  "messages": [{"role": "user", "content": "{INPUT}"}],
  "model": "gpt-4",
  "max_tokens": 500
}
```

Response Path (use nested notation):

```JSON
{
  "choices": [{
    "message": {"content": "{RESPONSE}"}
  }]
}
```

Phase 4: Authentication Handling

7. Token Management

Three common patterns:

A. Bearer Token (most common)

```Shell
curl -H "Authorization: Bearer sk-abc123..."
```

→ In Prisma AIRS: Add to "Custom Headers"

B. API Key Header

```Shell
curl -H "X-API-Key: your-key-here"
```

→ Add to Custom Headers with exact header name

C. Query Parameter

```Shell
curl "https://api.example.com/chat?api_key=abc123"
```

→ Include in endpoint URL

Phase 5: Testing & Validation

8. Test Request Chain

Test 1: Direct API call

```shell
curl -X POST https://api.example.com/v1/chat \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{"messages":[{"role":"user","content":"hello"}]}'
```

Test 2: Through your wrapper (if using)

```shell
curl -X POST http://your-wrapper:5006/v1/chat/completions \
  -H "Content-Type: application/json" \
  -d '{"messages":[{"role":"user","content":"hello"}]}'
```

Test 3: In Prisma AIRS

Use the UI "Test Target" button after configuration

Tools for Discovery

Tool	Use Case
Browser DevTools	Primary reconnaissance tool
Burp Suite / OWASP ZAP	For complex authentication flows
Postman	For testing/iterating on API calls
jq	For parsing JSON responses: `curl ...

Tool	Use Case
Wireshark	If API uses websockets or non-HTTP protocols

Common Gotchas

- **Rate Limiting:** Test endpoints may have different limits than production
- **Session Tokens:** Some apps use temporary JWT tokens that expire
- **CORS:** Browser-based discovery may show OPTIONS preflight requests
- **Streaming Responses:** Some AI APIs stream (SSE/WebSocket) - Prisma AIRS needs completed responses
- **Custom Headers:** Applications may require vendor-specific headers for routing

---

Real Example: Discovering a Custom App

Let's say you're testing "ChatWidget Pro" embedded in a website:

1. Open site with DevTools → Network tab
2. Send "hello" through chat widget
3. See API call: `POST https://api.chatwidget.pro/inference`
4. Request payload:

```JSON
{
  "input": "hello",
  "widget_id": "abc123",
  "config": {"temp": 0.7}
}
```

5. Response:

```JSON
{
  "output": "Hi there!",
  "tokens_used": 25
}
```

6. **Prisma AIRS config:**

Setting	Value
Endpoint	`https://api.chatwidget.pro/inference`
Request	`{"input": "{INPUT}", "widget_id": "abc123", "config": {"temp": 0.7}}`
Response path	`{"output": "{RESPONSE}"}`
Custom headers	`Authorization: Bearer [token from DevTools]`