Cloudturing

내부 사용자 인증 플러그인

외부 API 데이터 업로드 가이드

API Version: v1.0
인증 방식: HMAC-SHA256
데이터 형식: JSON
프로토콜: HTTPS

최종 업데이트: 2026-02-11
본 문서는 Cloudturing 내부 사용자 인증 플러그인의 외부 API 사용법을 안내합니다.

Contents

1 개요

Cloudturing 내부 사용자 인증 플러그인은 외부 시스템에서 API를 통해 사용자 데이터를 직접 업로드할 수 있는 기능을 제공합니다. 이를 통해 기존 시스템(CRM, ERP, HR 시스템 등)과의 연동이 가능합니다.

1.1 주요 특징

- **HMAC-SHA256 서명 기반 인증** — 안전한 API 호출 보장
- **일괄 업로드** — 전체 사용자 데이터를 한 번에 덮어쓰기
- **개별 추가** — 사용자를 한 명씩 추가
- **JSON 형식** — 표준 JSON 요청/응답
- **Replay Attack 방지** — 타임스탬프 기반 유효성 검증 (±5분)

1.2 사전 준비

API를 사용하기 위해서는 다음이 필요합니다:

1. Cloudturing 콘솔에 로그인
2. 챗봇 선택 → 플러그인 → 내부 사용자 인증
3. 데이터 소스 탭에서 **API Key 발급** 버튼 클릭
4. 발급된 **API Key**와 **Secret Key**를 안전하게 보관

중요

Secret Key는 발급 시 단 한 번만 표시됩니다. 반드시 안전한 곳에 복사하여 보관하세요. 분실 시 재발급이 필요하며, 기존 Key는 즉시 무효화됩니다.

2 인증 (Authentication)

모든 API 요청에는 3개의 인증 헤더가 필요합니다.

2.1 인증 헤더

헤더	타입	설명
`X-API-Key`	String	콘솔에서 발급받은 API Key
`X-Timestamp`	String	ISO 8601 형식의 현재 시각 (UTC)
`X-Signature`	String	HMAC-SHA256으로 생성한 서명

Table 1: 필수 인증 헤더

2.2 HMAC-SHA256 서명 생성

서명은 다음 단계로 생성합니다:

1. 현재 UTC 시간을 ISO 8601 형식으로 생성 (타임스탬프)
2. 서명 대상 문자열 조합: `{timestamp}.{requestBody}`
3. Secret Key를 키로 사용하여 HMAC-SHA256 해시 생성
4. 결과를 16진수(hex) 문자열로 변환

서명 대상 문자열 형식

`timestamp + "." + requestBody`

예시:

```
2026-01-15T09:30:00.000Z.{"users":[{"name":"홍길동",...}]}
```

타임스탬프 유효성

서버는 **±5분 이내**의 타임스탬프만 허용합니다. 이를 통해 Replay Attack을 방지합니다. 시스템 시계가 정확한지 확인하세요.

3 API 엔드포인트

기본 URL은 Cloudturing 콘솔 도메인에 따라 결정됩니다.

메서드	엔드포인트	용도	설명
POST	`/api/external/internal-users/bulk`	일괄 업로드	전체 사용자 데이터 덮어쓰기
POST	`/api/external/internal-users`	개별 추가	사용자 1명 추가

Table 2: API 엔드포인트 목록

3.1 일괄 업로드 (Bulk Upload)

기존 사용자 데이터를 **전체 덮어쓰기**합니다. 이 API를 호출하면 기존 데이터가 모두 삭제되고, 요청에 포함된 사용자 데이터로 교체됩니다.

3.1.1 요청 (Request)

필드	타입	필수	설명
`users`	Array	필수	사용자 객체 배열
`users[].name`	String	필수	사용자 이름
`users[].phone`	String	선택	전화번호
`users[].email`	String	선택	이메일 주소

Table 3: 일괄 업로드 요청 필드

```
POST /api/external/internal-users/bulk
Content-Type: application/json
X-API-Key: {YOUR_API_KEY}
X-Timestamp: 2026-01-15T09:30:00.000Z
X-Signature: {HMAC_SHA256_SIGNATURE}

{
  "users": [
    {
      "name": "홍길동",
      "phone": "010-1234-5678",
      "email": "hong@company.com"
    },
    {
      "name": "김영희",
      "phone": "010-9876-5432",
      "email": "kim@company.com"
    }
  ]
}
```

3.1.2 응답 (Response)

성공 (200 OK):

```
{
  "success": true,
  "message": "2명의 사용자 데이터가 업로드되었습니다.",
  "count": 2
}
```

실패 (4xx/5xx):

```
{
  "success": false,
  "message": "인증에 실패했습니다. API Key를 확인해주세요.",
  "code": "INVALID_API_KEY"
}
```

3.2 개별 추가 (Individual Add)

기존 데이터를 유지하면서 사용자 1명을 **추가**합니다.

3.2.1 요청 (Request)

필드	타입	필수	설명
`name`	String	필수	사용자 이름
`phone`	String	선택	전화번호
`email`	String	선택	이메일 주소

Table 4: 개별 추가 요청 필드

```
POST /api/external/internal-users
Content-Type: application/json
X-API-Key: {YOUR_API_KEY}
X-Timestamp: 2026-01-15T09:30:00.000Z
X-Signature: {HMAC_SHA256_SIGNATURE}

{
  "name": "홍길동",
  "phone": "010-1234-5678",
  "email": "hong@company.com"
}
```

3.2.2 응답 (Response)

성공 (201 Created):

```
{
  "success": true,
  "message": "사용자가 추가되었습니다.",
  "user": {
    "name": "홍길동",
```

```
    "phone": "010-****-5678",
    "email": "ho**@company.com"
  }
}
```

4 에러 코드

HTTP	코드	원인	해결 방법
400	`INVALID_REQUEST`	요청 형식 오류	JSON 형식과 필수 필드를 확인하세요
401	`INVALID_API_KEY`	잘못된 API Key	API Key가 정확한지 확인하세요
401	`INVALID_SIGNATURE`	서명 불일치	Secret Key와 서명 로직을 확인하세요
401	`EXPIRED_TIMESTAMP`	타임스탬프 만료	시스템 시계를 확인하세요 (±5분)
404	`CHATBOT_NOT_FOUND`	챗봇 없음	챗봇 UID와 API Key를 확인하세요
429	`RATE_LIMIT`	요청 한도 초과	잠시 후 다시 시도하세요
500	`INTERNAL_ERROR`	서버 오류	고객센터에 문의하세요

Table 5: 에러 코드 및 해결 방법

5 코드 샘플

아래는 일괄 업로드 API를 호출하는 각 언어별 코드 샘플입니다. 개별 추가의 경우 엔드포인트와 요청 본문만 변경하여 동일한 방식으로 사용할 수 있습니다.

5.1 cURL

```
DOMAIN="https://console.cloudturing.com"
PATH="/api/external/internal-users/bulk"

curl -X POST "${DOMAIN}${PATH}" \
  -H 'Content-Type: application/json' \
  -H 'X-API-Key: {YOUR_API_KEY}' \
  -H 'X-Timestamp: {ISO_8601_TIMESTAMP}' \
  -H 'X-Signature: {HMAC_SHA256_SIGNATURE}' \
  -d '{
  "users": [
    {
      "name": "홍길동",
      "phone": "010-1234-5678",
      "email": "hong@company.com"
    },
    {
      "name": "김영희",
      "phone": "010-9876-5432",
      "email": "kim@company.com"
    }
  ]
}'
```

5.2 Node.js

```
const crypto = require('crypto');

const API_KEY = '{YOUR_API_KEY}';
const SECRET_KEY = '{YOUR_SECRET_KEY}';
const DOMAIN = 'https://console.cloudturing.com';
const PATH = '/api/external/internal-users/bulk';
const ENDPOINT = DOMAIN + PATH;

const body = JSON.stringify({
  users: [
    {
      name: '홍길동',
      phone: '010-1234-5678',
      email: 'hong@company.com'
    },
    {
      name: '김영희',
      phone: '010-9876-5432',
      email: 'kim@company.com'
    }
  ]
});

const timestamp = new Date().toISOString();
const signature = crypto.createHmac('sha256', SECRET_KEY)
  .update(timestamp + '.' + body).digest('hex');

fetch(ENDPOINT, {
  method: 'POST',
  headers: {
    'Content-Type': 'application/json',
    'X-API-Key': API_KEY,
    'X-Timestamp': timestamp,
    'X-Signature': signature
  },
  body
})
  .then(res => res.json())
  .then(console.log);
```

5.3 Go

```go
package main

import (
    "crypto/hmac"
    "crypto/sha256"
    "encoding/hex"
    "fmt"
    "net/http"
    "strings"
    "time"
)

func main() {
    apiKey := "{YOUR_API_KEY}"
    secretKey := "{YOUR_SECRET_KEY}"
    domain := "https://console.cloudturing.com"
    path := "/api/external/internal-users/bulk"
    endpoint := domain + path

    body := `{
      "users": [
        {
          "name": "홍길동",
          "phone": "010-1234-5678",
          "email": "hong@company.com"
        },
        {
          "name": "김영희",
          "phone": "010-9876-5432",
          "email": "kim@company.com"
        }
      ]
    }`

    timestamp := time.Now().UTC().Format(time.RFC3339Nano)
    message := timestamp + "." + body

    mac := hmac.New(sha256.New, []byte(secretKey))
    mac.Write([]byte(message))
    signature := hex.EncodeToString(mac.Sum(nil))

    req, _ := http.NewRequest("POST", endpoint, strings.NewReader(body))
    req.Header.Set("Content-Type", "application/json")
    req.Header.Set("X-API-Key", apiKey)
    req.Header.Set("X-Timestamp", timestamp)
    req.Header.Set("X-Signature", signature)

    resp, _ := http.DefaultClient.Do(req)
    defer resp.Body.Close()
    fmt.Println(resp.Status)
}
```

5.4 Rust

의존성 (`Cargo.toml`):

```
[dependencies]
hmac = "0.12"
sha2 = "0.10"
hex = "0.4"
chrono = "0.4"
reqwest = { version = "0.11", features = ["json"] }
tokio = { version = "1", features = ["full"] }
```

```
use hmac::{Hmac, Mac};
use sha2::Sha256;
use chrono::Utc;
use reqwest::header::{HeaderMap, CONTENT_TYPE};

type HmacSha256 = Hmac<Sha256>;

#[tokio::main]
async fn main() -> Result<(), Box<dyn std::error::Error>> {
    let api_key = "{YOUR_API_KEY}";
    let secret_key = "{YOUR_SECRET_KEY}";
    let domain = "https://console.cloudturing.com";
    let path = "/api/external/internal-users/bulk";
    let endpoint = format!("{}{}", domain, path);

    let body = r#"{
      "users": [
        {
          "name": "홍길동",
          "phone": "010-1234-5678",
          "email": "hong@company.com"
        },
        {
          "name": "김영희",
          "phone": "010-9876-5432",
          "email": "kim@company.com"
        }
      ]
    }"#;

    let timestamp = Utc::now().to_rfc3339();
    let message = format!("{}.{}", timestamp, body);

    let mut mac = HmacSha256::new_from_slice(secret_key.as_bytes())?;
    mac.update(message.as_bytes());
    let signature = hex::encode(mac.finalize().into_bytes());

    let mut headers = HeaderMap::new();
    headers.insert(CONTENT_TYPE, "application/json".parse()?);
    headers.insert("X-API-Key", api_key.parse()?);
    headers.insert("X-Timestamp", timestamp.parse()?);
    headers.insert("X-Signature", signature.parse()?);
```

```
    let res = reqwest::Client::new()
        .post(&endpoint).headers(headers)
        .body(body.to_string())
        .send().await?;
    println!("{}", res.text().await?);
    Ok(())
}
```

5.5 PHP

```php
<?php

$apiKey = '{YOUR_API_KEY}';
$secretKey = '{YOUR_SECRET_KEY}';
$domain = 'https://console.cloudturing.com';
$path = '/api/external/internal-users/bulk';
$endpoint = $domain . $path;

$body = json_encode([
    'users' => [
        [
            'name' => '홍길동',
            'phone' => '010-1234-5678',
            'email' => 'hong@company.com'
        ],
        [
            'name' => '김영희',
            'phone' => '010-9876-5432',
            'email' => 'kim@company.com'
        ]
    ]
]);

$timestamp = gmdate('Y-m-d\TH:i:s.v\Z');
$message = $timestamp . '.' . $body;
$signature = hash_hmac('sha256', $message, $secretKey);

$ch = curl_init($endpoint);
curl_setopt_array($ch, [
    CURLOPT_POST => true,
    CURLOPT_POSTFIELDS => $body,
    CURLOPT_RETURNTRANSFER => true,
    CURLOPT_HTTPHEADER => [
        'Content-Type: application/json',
        'X-API-Key: ' . $apiKey,
        'X-Timestamp: ' . $timestamp,
        'X-Signature: ' . $signature
    ]
]);

$response = curl_exec($ch);
curl_close($ch);
echo $response;
```

5.6 Python

```python
import hmac, hashlib, json, requests
from datetime import datetime, timezone

API_KEY = '{YOUR_API_KEY}'
SECRET_KEY = '{YOUR_SECRET_KEY}'
DOMAIN = 'https://console.cloudturing.com'
PATH = '/api/external/internal-users/bulk'
ENDPOINT = DOMAIN + PATH

body = json.dumps({
    "users": [
        {
            "name": "홍길동",
            "phone": "010-1234-5678",
            "email": "hong@company.com"
        },
        {
            "name": "김영희",
            "phone": "010-9876-5432",
            "email": "kim@company.com"
        }
    ]
})

timestamp = datetime.now(timezone.utc).isoformat()
message = f"{timestamp}.{body}"
signature = hmac.new(
    SECRET_KEY.encode(), message.encode(), hashlib.sha256
).hexdigest()

response = requests.post(ENDPOINT, headers={
    "Content-Type": "application/json",
    "X-API-Key": API_KEY,
    "X-Timestamp": timestamp,
    "X-Signature": signature
}, data=body)

print(response.json())
```

5.7 Java

```
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import java.net.http.*;
import java.time.Instant;

public class ApiClient {
    private static String bytesToHex(byte[] bytes) {
        StringBuilder sb = new StringBuilder();
        for (byte b : bytes)
            sb.append(String.format("%02x", b));
        return sb.toString();
    }

    public static void main(String[] args) throws Exception {
        String apiKey = "{YOUR_API_KEY}";
        String secretKey = "{YOUR_SECRET_KEY}";
        String domain = "https://console.cloudturing.com";
        String path = "/api/external/internal-users/bulk";
        String endpoint = domain + path;

        String body = """
            {
              "users": [
                {
                  "name": "홍길동",
                  "phone": "010-1234-5678",
                  "email": "hong@company.com"
                },
                {
                  "name": "김영희",
                  "phone": "010-9876-5432",
                  "email": "kim@company.com"
                }
              ]
            }""";

        String timestamp = Instant.now().toString();
        String message = timestamp + "." + body;

        Mac mac = Mac.getInstance("HmacSHA256");
        mac.init(new SecretKeySpec(
            secretKey.getBytes(), "HmacSHA256"));
        String signature = bytesToHex(
            mac.doFinal(message.getBytes()));

        HttpRequest request = HttpRequest.newBuilder()
            .uri(URI.create(endpoint))
            .header("Content-Type", "application/json")
            .header("X-API-Key", apiKey)
            .header("X-Timestamp", timestamp)
            .header("X-Signature", signature)
            .POST(HttpRequest.BodyPublishers.ofString(body))
            .build();

        HttpResponse<String> response =
```

```
            HttpClient.newHttpClient().send(
                request,
                HttpResponse.BodyHandlers.ofString()
            );
        System.out.println(response.body());
    }
}
```

5.8 Ruby

```ruby
require 'net/http'
require 'json'
require 'openssl'
require 'time'

api_key = '{YOUR_API_KEY}'
secret_key = '{YOUR_SECRET_KEY}'
domain = 'https://console.cloudturing.com'
path = '/api/external/internal-users/bulk'
endpoint = domain + path

body = {
  users: [
    {
      name: '홍길동',
      phone: '010-1234-5678',
      email: 'hong@company.com'
    },
    {
      name: '김영희',
      phone: '010-9876-5432',
      email: 'kim@company.com'
    }
  ]
}.to_json

timestamp = Time.now.utc.iso8601(3)
message = "#{timestamp}.#{body}"
signature = OpenSSL::HMAC.hexdigest(
  'SHA256', secret_key, message
)

uri = URI(endpoint)
http = Net::HTTP.new(uri.host, uri.port)
http.use_ssl = true

req = Net::HTTP::Post.new(uri)
req['Content-Type'] = 'application/json'
req['X-API-Key'] = api_key
req['X-Timestamp'] = timestamp
req['X-Signature'] = signature
req.body = body

res = http.request(req)
puts res.body
```

5.9 C#

```
using System.Security.Cryptography;
using System.Text;

var apiKey = "{YOUR_API_KEY}";
var secretKey = "{YOUR_SECRET_KEY}";
var domain = "https://console.cloudturing.com";
var path = "/api/external/internal-users/bulk";
var endpoint = domain + path;

var body = """
  {
    "users": [
      {
        "name": "홍길동",
        "phone": "010-1234-5678",
        "email": "hong@company.com"
      },
      {
        "name": "김영희",
        "phone": "010-9876-5432",
        "email": "kim@company.com"
      }
    ]
  }""";

var timestamp = DateTime.UtcNow.ToString("o");
var message = $"{timestamp}.{body}";
using var hmac = new HMACSHA256(
    Encoding.UTF8.GetBytes(secretKey));
var signature = Convert.ToHexString(
    hmac.ComputeHash(Encoding.UTF8.GetBytes(message))
).ToLower();

using var client = new HttpClient();
var request = new HttpRequestMessage(
    HttpMethod.Post, endpoint);
request.Content = new StringContent(
    body, Encoding.UTF8, "application/json");
request.Headers.Add("X-API-Key", apiKey);
request.Headers.Add("X-Timestamp", timestamp);
request.Headers.Add("X-Signature", signature);

var response = await client.SendAsync(request);
Console.WriteLine(
    await response.Content.ReadAsStringAsync());
```

6 자주 묻는 질문 (FAQ)

6.1 API Key와 Secret Key의 차이는 무엇인가요?

- **API Key**: 요청을 보낸 클라이언트를 식별하는 공개 키입니다. 요청 헤더에 포함됩니다.
- **Secret Key**: 서명을 생성하는 데 사용되는 비밀 키입니다. **절대 외부에 노출하지 마세요.**

6.2 일괄 업로드 시 기존 데이터는 어떻게 되나요?

일괄 업로드(`/bulk`) API는 **전체 덮어쓰기** 방식입니다. 기존에 등록된 모든 사용자 데이터가 삭제되고, 요청에 포함된 데이터로 완전히 교체됩니다. 기존 데이터를 유지하면서 추가하려면 개별 추가 API를 사용하세요.

6.3 서명 오류가 발생합니다. 어떻게 해결하나요?

다음 사항을 확인하세요:

1. **Secret Key**가 정확한지 확인
2. **타임스탬프**가 UTC 기준 ISO 8601 형식인지 확인
3. 서명 대상 문자열이 `{timestamp}.{requestBody}` 형식인지 확인
4. 요청 본문(body)이 서명 생성 시 사용한 것과 **완전히 동일**한지 확인 (공백, 줄바꿈 포함)
5. 시스템 시계가 정확한지 확인 (±5분 허용)

6.4 요청 제한(Rate Limit)이 있나요?

네, 과도한 요청을 방지하기 위해 Rate Limit이 적용됩니다. 한도를 초과하면 `429 Too Many Requests` 응답이 반환됩니다. 잠시 후 다시 시도하세요.