

# 인증 코드 부여 흐름 구축 및 문제 해결 - OAuth 개선 사항: Cisco Collaboration 솔루션 12.0

## 목차

[소개](#)

[사전 요구 사항](#)

[요구 사항](#)

[사용되는 구성 요소](#)

[배경 정보](#)

[주요 기능](#)

[중요 고려 사항](#)

[권한 부여 코드 부여 플로우의 요소](#)

[구성](#)

[네트워크 다이어그램](#)

[토큰 새로 고침](#)

[새로 고침 토큰 취소](#)

[다음을 확인합니다.](#)

[문제 해결](#)

[관련 정보](#)

## 소개

이 문서에서는 특히 모바일의 Jabber에 대해 다양한 디바이스에서 Jabber 사용자 환경을 개선하기 위해 새로 고침 토큰을 기반으로 권한 부여 코드 부여 플로우가 실행되는 방법에 대해 설명합니다.

## 사전 요구 사항

### 요구 사항

다음 주제에 대한 지식을 보유하고 있으면 유용합니다.

- Cisco CUCM(Unified Communications Manager) 12.0 버전
- SSO(Single Sign On)/SAML
- Cisco Jabber
- Microsoft ADFS
- ID 공급자(IdP)

이러한 항목에 대한 자세한 내용은 다음 링크를 참조하십시오.

- [Cisco Unified Communications용 SAML SSO 구축 설명서](#)
- [Unified Communications Manager SAML SSO 구성 예:](#)
- [SAML SSO 컨피그레이션을 위한 AD FS 버전 2.0 설정 예:](#)

### 사용되는 구성 요소

이 문서의 정보는 다음 소프트웨어를 기반으로 합니다.

- Microsoft ADFS(IdP)
- LDAP Active Directory
- Cisco Jabber 클라이언트
- CUCM 12.0

이 문서의 정보는 특정 랩 환경의 디바이스를 토대로 작성되었습니다. 이 문서에 사용된 모든 디바이스는 초기화된(기본) 컨피그레이션으로 시작되었습니다. 네트워크가 작동 중인 경우 모든 명령의 잠재적인 영향을 이해해야 합니다.

## 배경 정보

현재 인프라와 함께 Jabber SSO 흐름은 CUCM Authz 서비스에서 짧은 수명 액세스 토큰을 할당하는 Implicit Grant Flow를 기반으로 합니다.

액세스 토큰 만료 후 CUCM은 재인증을 위해 Jabber를 IdP로 리디렉션합니다.

이로 인해 사용자 환경이 좋지 않습니다. 특히 모바일 Jabber에서 사용자에게 자격 증명을 자주 입력하라는 메시지가 표시됩니다.

또한 Security Re-architecture Solution은 SSO 및 비 SSO 시나리오에 대해 Jabber 및 엔드포인트 로그인 흐름을 통합하기 위한 Refresh Tokens(엔드포인트/기타 협업 앱으로 확장 가능) 접근 방식을 사용하여 권한 부여 코드 부여 흐름을 제안합니다.

## 주요 기능

- 권한 부여 코드 부여 흐름은 다양한 장치, 특히 모바일 Jabber의 경우 Jabber 사용자 환경을 개선하기 위해 새로 고침 토큰(엔드포인트/기타 협업 앱으로 확장 가능)을 기반으로 합니다.
- 다양한 협업 애플리케이션이 클라이언트 리소스 요청을 검증하고 응답할 수 있도록 자체 포함된 서명 및 암호화된 OAuth 토큰을 지원합니다.
- 암시적 부여 흐름 모델은 이전 버전과의 호환성을 허용하는 상태로 유지됩니다. 이렇게 하면 권한 부여 코드 부여 플로우로 이동하지 않은 다른 클라이언트(예: RTMT)에 대한 원활한 경로가 허용됩니다.

## 중요 고려 사항

- 기존 jabber 클라이언트가 새 CUCM에서 작동할 수 있도록 구현합니다(암시적 부여 및 권한 부여 코드 부여 흐름을 모두 지원하므로). 또한 새 jabber는 기존 CUCM에서 작동할 수 있습니다. Jabber는 CUCM이 권한 부여 코드 부여 흐름을 지원하는지 여부를 확인하고 이 모델을 지원하는 경우에만 이 모델을 전환하고 암시적 부여 흐름을 사용합니다.
- AuthZ 서비스는 CUCM 서버에서 실행됩니다.
- AuthZ는 암시적 부여 흐름만 지원합니다. 이는 새로 고침 토큰/오프라인 액세스 토큰이 없음을 의미합니다. 클라이언트가 새 액세스 토큰을 원할 때마다 사용자는 IdP로 다시 인증해야 합니다.
- 액세스 토큰은 구축이 SSO가 활성화된 경우에만 발급되었습니다. 이 경우 비 SSO 구축이 작동하지 않았고 액세스 토큰이 모든 인터페이스에서 일관성 있게 사용되지 않았습니다.
- Access Tokens(액세스 토큰)는 자체 포함된 것이 아니라 발급된 서버의 메모리에 보존됩니다. CUCM1에서 액세스 토큰을 발급한 경우 CUCM1에서만 확인할 수 있습니다. 클라이언트가

CUCM2에서 서비스에 액세스하려고 할 경우 CUCM2는 CUCM1에서 해당 토큰을 검증해야 합니다. 네트워크 지연(프록시 모드).

- 사용자가 IdP를 사용하여 재인증할 때 영숫자 키패드에서 자격 증명을 다시 입력해야 하므로 모바일 클라이언트의 사용자 환경은 매우 좋지 않습니다(일반적으로 몇 가지 요소에 따라 1시간에서 8시간으로 실행).
- 여러 인터페이스를 통해 여러 애플리케이션과 통신하는 클라이언트는 여러 자격 증명/블록을 유지해야 합니다. 2개의 유사한 클라이언트에서 동일한 사용자 로그인을 원활하게 지원하지 않습니다. 예를 들어 사용자 A는 서로 다른 2개의 iPhone에서 실행되는 jabber 인스턴스에서 로그인합니다.
- AuthZ - SSO 및 비 SSO 구축을 모두 지원합니다.
- AuthZ - 암시적 권한 부여 흐름 + 권한 부여 코드 부여 흐름을 지원합니다. 이전 버전과 호환되므로 RTMT와 같은 고객은 적응할 때까지 작업을 계속할 수 있습니다.
- Authorization code grant flow를 사용하면 AuthZ는 액세스 토큰 및 새로 고침 토큰을 발급합니다. 새로 고침 토큰을 사용하여 인증 없이 다른 액세스 토큰을 가져올 수 있습니다.
- 액세스 토큰은 자체 포함, 서명 및 암호화되며 JWT(JSON 웹 토큰) 표준(RFC 규격)을 사용합니다.
- 서명 및 암호화 키는 클러스터에 공통적입니다. 클러스터의 모든 서버에서 액세스 토큰을 확인할 수 있습니다. 메모리에 유지할 필요가 없습니다.
- CUCM 12.0에서 실행되는 서비스는 클러스터의 중앙 집중식 인증 서버입니다.
- 새로 고침 토큰은 데이터베이스(DB)에 저장됩니다. 관리자는 필요한 경우 이를 취소할 수 있어야 합니다. 취소는 사용자 ID 또는 사용자 ID 및 클라이언트 ID를 기반으로 합니다.
- 서명된 액세스 토큰을 사용하면 다른 제품에서 액세스 토큰을 저장할 필요 없이 검증할 수 있습니다. 구성 가능한 액세스 토큰 및 새로 고침 토큰 수명(각각 1시간 및 60일 기본값)
- JWT 형식은 Spark와 연계되어 향후에 Spark Hybrid 서비스와 시너지를 가능하게 합니다.
- 2개의 유사한 디바이스에서 동일한 사용자 로그인을 지원합니다. 예: 사용자 A는 서로 다른 2개의 iPhone에서 실행되는 jabber 인스턴스에서 로그인할 수 있습니다.

## 권한 부여 코드 부여 플로우의 요소

- 인증 Z 서버
- 암호화 키
- 서명 키
- 토큰 새로 고침

## 구성

이 기능은 기본적으로 활성화되어 있지 않습니다.

1단계. 이 기능을 활성화하려면 시스템 > 엔터프라이즈 매개변수로 이동합니다.

2단계. 이미지에 표시된 대로 Refresh Login Flow(로그인 흐름 새로 고침)가 있는 매개변수 OAuth를 Enabled(활성화됨)로 설정합니다.

SSO and OAuth Configuration		
OAuth Access Token Expiry Timer (minutes) *	<input type="text" value="60"/>	60
OAuth Refresh Token Expiry Timer (days) *	<input type="text" value="60"/>	60
Redirect URIs for Third Party SSO Client	<input type="text"/>	
SSO Login Behavior for iOS *	Use embedded browser (WebView) ▼	Use embedded browser (WebView)
OAuth with Refresh Login Flow *	Enabled ▼	Disabled
Use SSO for RTMT *	True ▼	True

- 액세스 토큰은 서명되고 암호화됩니다. 서명 및 암호화 키는 클러스터에 공통적입니다. 이는 클러스터의 모든 노드가 액세스 토큰을 검증할 수 있음을 의미합니다.
- 액세스 토큰은 JWT 형식(RFC 7519)입니다.
- 액세스 토큰은 이전 토큰 및 새 토큰 형식 모두에 적용할 수 있는 엔터프라이즈 매개 변수 (OAuth 액세스 토큰 만료 타이머)를 다시 사용합니다.
- 기본값 - 60분
- 최소값 - 1분
- 최대값 - 1440분

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCIsImtpZCI6IjhhMGQ1MzI0LWY0ZjAtNGIwYi04MTFlLTRhNTlmZGI2YjcyMjppj
Mjc3MGM5N2JkYTlkMzRmZDA1YTdlYTFhZQWZTU0Y2E4MGJkZDdlZTM1ZDk3MDNiNjBiNTQ5MTBiZDQ0ODRiIn0.eyJwcm12
YXR1IjoizXlKaGJHY2lPaUprYVhJaUxDSmpkSGtpT2lKS1YxUWlMQ0psYm1NaU9pSkJNVEk0UTBKRExVaFRNaUySWl3aWEy
bGtJam9pT0dRd1pEVXpNalF0WmpSbU1DMDBZakJpTFRneE1XVXROR0UxT1daa1lqWmlOek15T21Vd1ptUm1ZMk16WlRRMU5E
RTFOV0ZpTkrJek5tRTJOMlV4T0RChU1qWmxZMk13WXpJee56SX1OREJtWlRFellXWX1Oak14TkRkalpHVXpNR1l3TjJJaWZR
Li5xQWd6aGdRaTVMkdlaDl5V2RvN25nLmdMTHNpaTRjQk50c1NEUXRjTE51RWRnWTl4WkVJczJ4YzBaeTFGQjZQNmNzWWJf
ZkRnaDRZby04V1NaNjUzdXowbnFOalpXT1E1dGdnYW9qMlp6ZFk2ZzN2SWFHbF9JWUtNdKNIWwNscmt4YUFGTk5MWExLQ1Jm
aTA2LVk2V3l1dUdxNmpNwK5DbnlKX1pTbUpkVFQwc1Z4RTdGTXVxaUJSMElrRGdyVDDvOFNXMEY5cXFadndEZDJSaDdqNkRJ
WGdks3VtOwlTU2xNU1pjejhueVdic01Udk5yMWY0M25VenJzMHk5WwN6NnBDX0czZmlwYjJ5X2VWLVFkcFh4TUo2bnZodXcy
djRiUGVkm3VMQlpaVWl0Q3B6TUVDdW5NM1h1TVBrTGdlS1NqWG44aGhPRFNVcWlWQ0Uta3RzdnRbc2Q0RnJxcGNxWlZiS0Zi
VTFRbU0w2pMYVJtUk9IVl1lQVkc0a3FBdTRWalVMUzVCRWszNnZ4Nmp3U3BMUy1IdTcwbVRNcmR3dmV5Q2ZOYkhyT0FlVmVv
ekFIR3JqdGhmaFpmSFVUTWZiNkMtX2tOQVJGQWdDclZTZy0wUzlxblJvTWVvKUEuNETEE4MDJiaWwtNDJjOC15Mwo4X1FVaC02
UUtCV2dodVd4VWtBODRpekFFaWl0QTlsSHFKM3Nxd2JFNURkZmhIay05bTJfTTN5MwVlWVkdORVQ3ZW9XVDBqWllnRGRBQjFz
UGwxLTLafSNYYmsydTE3SkJVRV9FOXIOV0tWmNBqWgtiN0lQSWgtQ3JWQTZkcVdQRHVIbmx1V19wblNlYnYtTkZVbGQ0WEY3
cmZLYmQySlg4eUhhX05pOVVVUnUwZVdsNwXGRUVabklubmFKZEHLUZrb3VuN2xHSFlwSE4ydXVudmRnOHZVZzZsa0JPbmoz
eUFjclZTMGxKc1NwdUxYF1dwd2c4YjdBdDM3d3AtMwt2Y1ZQaWpCQ11CV181d2JzbTFYd2k4MVC2WHVpNzZmZmVg3cEJVQnBf
T2VRNzQ2ZXJJekNUUFZCYUpZUGJuzWEtdFhsU3RmZzBGEVRmbnbnX1Vzazl3QXJkeme4c204T0FQaWmxZmFQOG0uUTdFN0FV
X2xUVnNmZFI2bnkydUdhQSJ9.u2fJrVA55NQC3esPb4kcodt5rnjcl0-5uEDdUf-
KnCYEPBZ7t2CTsMMVVE3nfrhm39MfTlNS-qVOVpuoW_51NYaENXQMxfxlU9aXp944QiU1OeFQKj_g-
n2dEINRStbtUc3KMKqtz38BFf1g2Z51sdlnBn4XyVVPgGCf4XSfsFIa9fF051awQ0LcCv6YQGer_6nk7t6F1MzPzBzZja1a
bpm--6LNSzjPftEiexpD2oXvW8V10Z9ggNk5Pn3Ne4RzqK09J9WChaJSXkTTE5G39EZcePmVntcbayq-
L2pAK5weDa2k4uYmfAQAwcTOhUrwK3yilwqjHAamcG-CoipZQ
```

OAuth Refresh Token Expiry Timer" parameter in enterprise parameters page in CUCM.  
 Path: System -> Enterprise parameters  
 Values are integers ranging from 1 - 90  
 Minimum lifetime = 1 Day  
 Default lifetime = 60 days  
 Maximum lifetime = 90 days

클라이언트가 요청을 할 때마다 새 액세스 토큰이 발급됩니다. 기존 항목은 다음과 같은 기간 동안 계속 유효합니다.

- 서명/암호화 키가 변경되지 않았습니다.
- 유효성(토큰 내에 저장됨)이 중단됩니다.
- JSON 웹 토큰: 세 부분으로 구성되며 점으로 구분됩니다. 헤더, 페이로드 및 서명.

샘플 액세스 토큰:

- 굵게 강조 표시된 토큰의 시작 부분에 헤더가 있습니다.
- 중간 부분은 페이로드입니다.
- 끝에 굵게 강조 표시된 토큰은 Signature입니다.

## 네트워크 다이어그램

관련된 통화 흐름에 대한 개괄적인 개요를 소개합니다.



칩 토큰의 모든 새로 고침 토큰을 취소할 수 있습니다.

사용자에 대한 디바이스 기반 RT를 취소하려면

- client\_id abc로 식별된 사용자 xyz 및 디바이스에 대한 RT를 취소합니다.
- [https://cucm-193:8443/ssosp/token/revoke?user\\_id=xyz&client\\_id=abc](https://cucm-193:8443/ssosp/token/revoke?user_id=xyz&client_id=abc)

서명 및 암호화 키




- 서명 키는 공개/개인 키 쌍을 가진 RSA 기반입니다.
- 암호화 키는 대칭 키입니다.
- 이러한 키는 게시자에서만 생성되며 클러스터의 모든 노드에 배포됩니다.
- 나열된 옵션을 사용하여 서명 키와 암호화 키를 모두 다시 생성할 수 있습니다. 그러나 이 작업은 키가 손상되었다고 관리자가 믿는 경우에만 수행해야 합니다. 이러한 키 중 하나를 재생성하면 AuthZ 서비스에서 발급한 모든 액세스 토큰이 유효하지 않게 됩니다.
- 서명 키는 UI 및 CLI를 사용하여 다시 생성할 수 있습니다.
- 암호화 키는 CLI에서만 재생성할 수 있습니다.

CUCM의 **Cisco Unified OS Administration** 페이지에서 Authz 인증서(서명 키)의 재생성은 이미지와 같습니다.

Certificate Details(Self-signed) - Internet Explorer provided by Cisco Systems, Inc.


https://10.77.29.184/cmplatform/certificateEdit.do?cert=/usr/local/platform/.security/authz/certs/authz.j ✖ Certificate error

### Certificate Details for AUTHZ\_CUCM-184, authz

 Regenerate
  Download .PEM File
  Download .DER File

---

**Status**

 Status: Ready

---

**Certificate Settings**

File Name	authz.pem
Certificate Purpose	authz
Certificate Type	certs
Certificate Group	product-cpi
Description(friendly name)	Self-signed certificate generated by system

---

**Certificate File Data**

```

[
[
Version: V3
Subject: L=i, ST=i, CN=AUTHZ_CUCM-184, OU=i, O=i, C=IN
Signature Algorithm: SHA256withRSA, OID = 1.2.840.113549.1.1.11

Key: CiscoJ RSA Public Key, 2048 bits
modulus:
310088952412132774650041525392629167237879710935753621934671843
216346326898490353644164813514840735197164588955185219996734516
256663568507413849247845292675452179850077675141884383314726763
520023902784651553941826511494962731151521090167892375623419501
739811988911210916820812069748957615302991414362015465824669063
319779866264424936428249029193098223306846888723560182717860238
318402233050626785154245146789308145325775236137097363983609689
  
```

CLI 명령을 사용하여 Authz 서명 키를 재생성하는 방법은 이미지에 표시된 것과 같습니다.

```
CUCM-184 login: admin
Password:
Last login: Tue Nov 15 15:43:52 on tty1
Command Line Interface is starting up, please wait ...
```

```
Welcome to the Platform Command Line Interface
```

```
VMware Installation:
 1 vCPU: Intel(R) Xeon(R) CPU E5-2643 0 @ 3.30GHz
Disk 1: 80GB, Partitions aligned
6144 Mbytes RAM
```

```
admin:set ke
admin:set key regen authz signing
```

```
WARNING: This operation will regenerate the Authorization Service signing key and restart the Authorization Service on all the nodes. It is recommended that this command be run off-hours to avoid end user impact.
```

```
Proceed with regeneration (yes/no)? yes
```

```
signing key for the Authorization service generated successfully.
```

```
admin:_
```

관리자는 CLI를 사용하여 인증 서명 및 암호화 키를 표시할 수 있습니다. 키의 해시가 원래 키가 아니라 표시됩니다.

키를 표시하는 명령:

서명 키: **show key authz signing**과 이미지에 표시된 것처럼 표시합니다.

```
admin:show key authz signing
authz signing key with checksum: a155d81be734850226f990a62816f1ae last synced on: 06/09/2017 13:04:47
```

암호화 키: **show key authz encryption**과 이미지에 표시된 것처럼 표시합니다.

```
admin:show key authz encryption
authz encryption key with checksum: 88edce92173e33f9cedbbfb09cd0e8c4 last synced on: 06/14/2017 16:22:06
```

**참고:** 서명 인증과 암호화 인증은 항상 다릅니다.

## 다음을 확인합니다.

이 섹션을 사용하여 컨피그레이션이 제대로 작동하는지 확인합니다.

CUC(Cisco Unity Connection) 서버에서 OAuth를 사용하려면 네트워크 관리자가 두 단계를 수행해야 합니다.

1단계. CUCM에서 OAuth 토큰 서명 및 암호화 키를 가져오도록 Unity Connection 서버를 구성합니다.

2단계. CUC 서버에서 OAuth 서비스를 활성화합니다.



**참고:** 서명 및 암호화 키를 가져오려면 CUCM 호스트 세부 정보 및 CUCM AXL 액세스에서 활성화된 사용자 계정으로 Unity를 구성해야 합니다. 이 구성이 구성되지 않은 경우 Unity Server는 CUCM에서 OAuth 토큰을 검색할 수 없으며 사용자의 음성 메일 로그인을 사용할 수 없습니다.

Cisco Unity Connection Administration(Cisco Unity Connection 관리) > System Settings(시스템 설정) > Authz Servers(인증 서버)로 이동합니다.

**New Authz Server**

Authz Servers   Reset   Help

**New Authz Server**

Display Name\*

Authz Server\*

Port\*

Username\*

Password\*

Ignore Certificate Errors

Fields marked with an asterisk (\*) are required.

## 문제 해결

이 섹션에서는 컨피그레이션 문제를 해결하는 데 사용할 수 있는 정보를 제공합니다.

**참고:** OAuth가 사용되고 Cisco Jabber 사용자가 로그인할 수 없는 경우 항상 CUCM 및 IM&P(Instant Messaging and Presence) 서버에서 서명 및 암호화 키를 검토합니다.

네트워크 관리자는 모든 CUCM 및 IM&P 노드에서 다음 두 명령을 실행해야 합니다.

- 키 인증 서명 표시
- `show key authz encryption`

서명 인증 및 암호화 인증 출력이 모든 노드에서 일치하지 않을 경우 재생성해야 합니다. 이를 수행하려면 다음 두 명령을 모든 CUCM 및 IM&P 노드에서 실행해야 합니다.

- `key regen authentication` 암호화 설정
- 키 `regen authin` 서명 설정

그런 다음 Cisco Tomcat 서비스를 모든 노드에서 다시 시작해야 합니다.

키 불일치와 함께 이 오류 행은 Cisco Jabber 로그에서 찾을 수 있습니다.

```
2021-03-30 14:21:49,631 WARN [0x0000264c] [vices\impl\system\SingleSignOn.cpp(1186)] [Single-Sign-On-Logger] [CSFUnified::SingleSignOn::Impl::handleRefreshTokenFailure] - Failed to get valid access token
```

from refresh token, maybe server issue.

sso 앱 로그는 다음 위치에 생성됩니다.

- **파일 보기** `activelog platform/log/ssoApp.log` 로그 수집을 위해 추적 구성이 필요하지 않습니다. SSO 앱 작업이 완료될 때마다 새 로그 항목이 `ssoApp.log` 파일에 생성됩니다.
- **SSOSP 로그: 파일 목록** `activelog tomcat/logs/ssosp/log4j`  
sso가 활성화될 때마다 이 위치에 `ssosp00XXX.log`라는 새 로그 파일이 생성됩니다. 다른 모든 SSO 작업 및 모든 OAuth 작업도 이 파일에 기록됩니다.
- **인증서 로그: 파일 목록** `활성 플랫폼/로그/certMgmt*.log`  
AuthZ 인증서가 재생성될 때마다(UI 또는 CLI) 이 이벤트에 대해 새 로그 파일이 생성됩니다.  
authz 암호화 키 재생성의 경우 이 이벤트에 대해 새 로그 파일이 생성됩니다.

## 관련 정보

[Cisco Collaboration Solution 릴리스 12.0으로 OAuth 구축](#)