Comprendere il sistema CORS (Cross-Origin Resource Sharing) per Finesse

Sommario

Introduzione

Questo documento descrive in modo completo la condivisione delle risorse tra le origini in modo che, durante la risoluzione dei problemi, i processi sottostanti siano perfettamente compresi.

Prerequisiti

Requisiti

Cisco raccomanda la conoscenza dei seguenti argomenti:

- Cisco Unified Contact Center Enterprise (UCCE) release 12.6.X
- Cisco Packaged Contact Center Enterprise (PCCE) release 12.6.X
- Cisco Finesse release 12.6.X
- Cisco Unified Intelligence Center (CUIC) release 12.6.X

Componenti usati

Le informazioni fornite in questo documento si basano sulle seguenti versioni software e hardware:

- UCCE release 12.6.2
- Finesse release 12.6.2
- CUIC release 12.6.2

Le informazioni discusse in questo documento fanno riferimento a dispositivi usati in uno specifico ambiente di emulazione. Su tutti i dispositivi menzionati nel documento la configurazione è stata

ripristinata ai valori predefiniti. Se la rete è operativa, valutare attentamente eventuali conseguenze derivanti dall'uso dei comandi.

Premesse

Che cos'è CORS

Il sistema CORS (Cross-Origin Resource Sharing) consente ai server di controllare a quali siti Web (domini, protocolli e porte) è consentito l'accesso alle risorse. Mentre i browser normalmente bloccano le richieste provenienti da origini diverse (la stessa regola di origine), CORS offre ai server la possibilità di allentare selettivamente questa restrizione. In pratica, i server utilizzano speciali intestazioni HTTP per indicare al browser quali origini sono consentite, quali tipi di richieste sono consentiti (come GET, POST e così via) e quali intestazioni personalizzate possono essere incluse. Questo consente ai server di decidere chi può accedere alle API e come, dall'accesso completamente aperto a quello strettamente limitato. CORS funziona facendo in modo che il browser e il server comunichino attraverso queste intestazioni HTTP per gestire le richieste tra origini.

CORS utilizza le intestazioni HTTP per abilitare le richieste tra origini controllate. Il browser e il server comunicano tramite queste intestazioni, con il server che specifica le origini, i metodi e le intestazioni consentite. Se le intestazioni di risposta del server sono mancanti o non valide, il browser blocca la risposta, applicando lo stesso criterio di origine. Per alcune richieste, il browser invia prima una richiesta di verifica preliminare al server per assicurarsi che accetti la richiesta di verifica incrociata effettiva.

I browser utilizzano le richieste di verifica preliminare per verificare se un server consente una richiesta di origine incrociata prima di inviare la richiesta reale. Queste richieste di verifica preliminare includono dettagli quali il metodo HTTP e le intestazioni personalizzate. I server basati su CORS possono quindi rispondere, autorizzando o negando la richiesta effettiva. Se un server non è configurato per CORS, non risponde correttamente alla verifica preliminare e il browser blocca la richiesta effettiva, proteggendo in tal modo il server da accessi incrociati indesiderati.

La condivisione delle risorse (CORS) è fondamentale per la sicurezza e la funzionalità del Web. Consente l'accesso controllato a risorse di origini diverse (domini, protocolli, porte), necessario perché i browser applicano un criterio di origine identica che normalmente blocca tale accesso.

Ciclo di vita di un CORS

Una richiesta CORS è composta da due parti: il client che effettua la richiesta e il server che la riceve. Sul lato client, lo sviluppatore scrive il codice JavaScript per inviare la richiesta al server. Il server risponde alla richiesta impostando intestazioni specifiche CORS speciali per indicare che la richiesta di origine incrociata è consentita. Senza la partecipazione sia del client che del server, la richiesta CORS non riesce.

Gli elementi chiave di una richiesta CORS sono il client, il browser e il server. Il client richiede alcuni dati dal server, ad esempio una risposta API JSON o il contenuto di una pagina Web. Il browser funge da intermediario attendibile per verificare che il client possa accedere ai dati dal

server.

Client:

Il client è un frammento di codice JavaScript in esecuzione su un sito Web ed è responsabile dell'avvio della richiesta CORS

Nota: Finesse è un'applicazione Web. Viene installato su un server e gli agenti vi accedono semplicemente utilizzando i browser Web, eliminando la necessità di installazioni sul lato client o di manutenzione dei plug-in o di altro software. Come dimostrato nell'esempio di CORS in Action con Cisco Finesse, questa architettura supporta funzionalità quali i report di dati in tempo reale. In questo contesto, il codice JavaScript del gadget Cisco Finesse Live Data agisce come client, mentre Cisco CUIC funge da server nell'ambito del ciclo di vita CORS. Essenzialmente, il client Finesse basato su browser interagisce con il server CUIC per recuperare i dati in tempo reale.

Client o utente:

A volte le parole client e user sono utilizzate in modo intercambiabile, ma sono diverse nel contesto di CORS. Un utente è una persona che visita un sito Web o un utente Finesse (agente o supervisore) che accede a Finesse in questo contesto, mentre un client è il codice effettivamente servito da quel sito Web. Più utenti possono visitare lo stesso sito Web e ricevere lo stesso codice client JavaScript.

Browser:

Il browser, noto anche come agente utente, ospita il codice lato client. Svolge un ruolo cruciale in CORS aggiungendo ulteriori informazioni alle richieste in uscita, consentendo al server di identificare il client. Inoltre, il browser interpreta la risposta del server, determinando se consegnare i dati al client o restituire un errore. Queste azioni lato browser sono essenziali per mantenere la protezione fornita dalla stessa regola di origine. Senza l'applicazione delle regole CORS da parte del browser, i client potrebbero effettuare richieste non autorizzate, compromettendo questo meccanismo di sicurezza vitale.

Server:

Il server è la destinazione della richiesta CORS ed è CUIC per esempio di gadget Live Data con Cisco Finesse. Il server memorizza i dati desiderati dal client e ha l'ultima possibilità di stabilire se la richiesta CORS è consentita o meno.

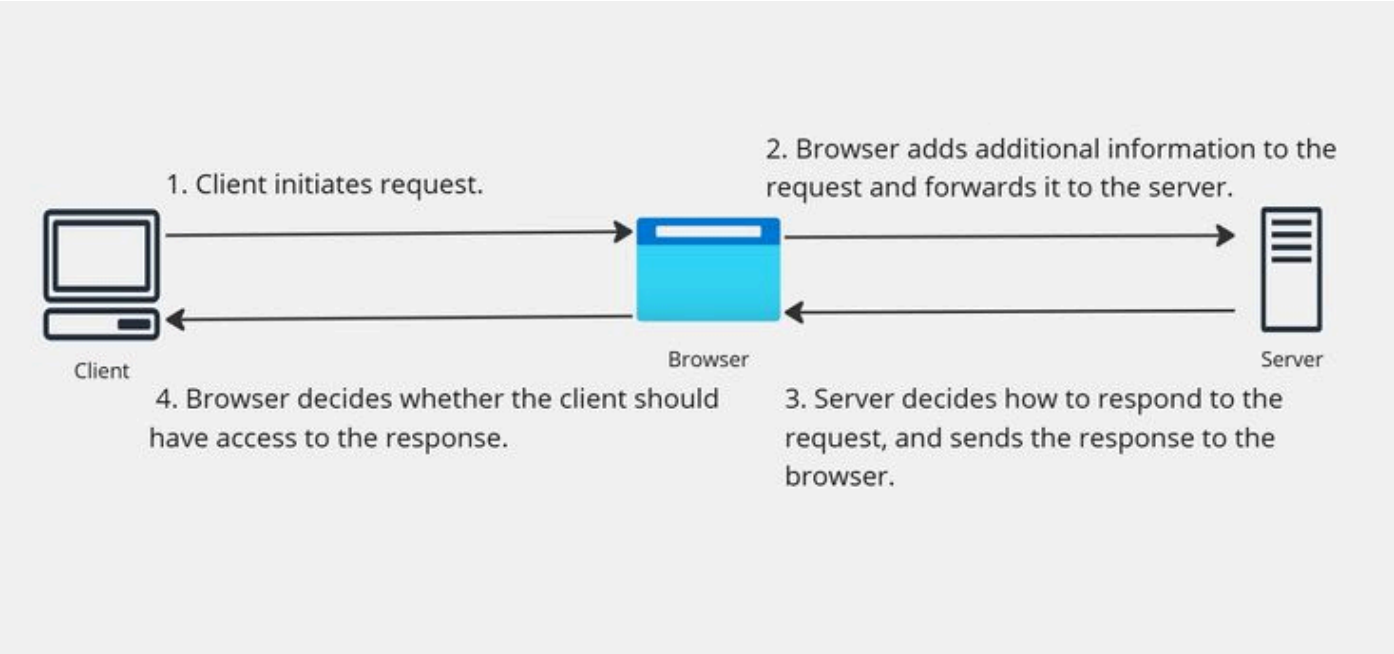
Ora che sapete chi è coinvolto in una richiesta CORS, diamo uno sguardo a come tutti lavorano insieme. Le immagini seguenti illustrano il ciclo di vita CORS di alto livello:

1. Il client avvia la richiesta.

2. Il browser aggiunge ulteriori informazioni alla richiesta e le inoltra al server.

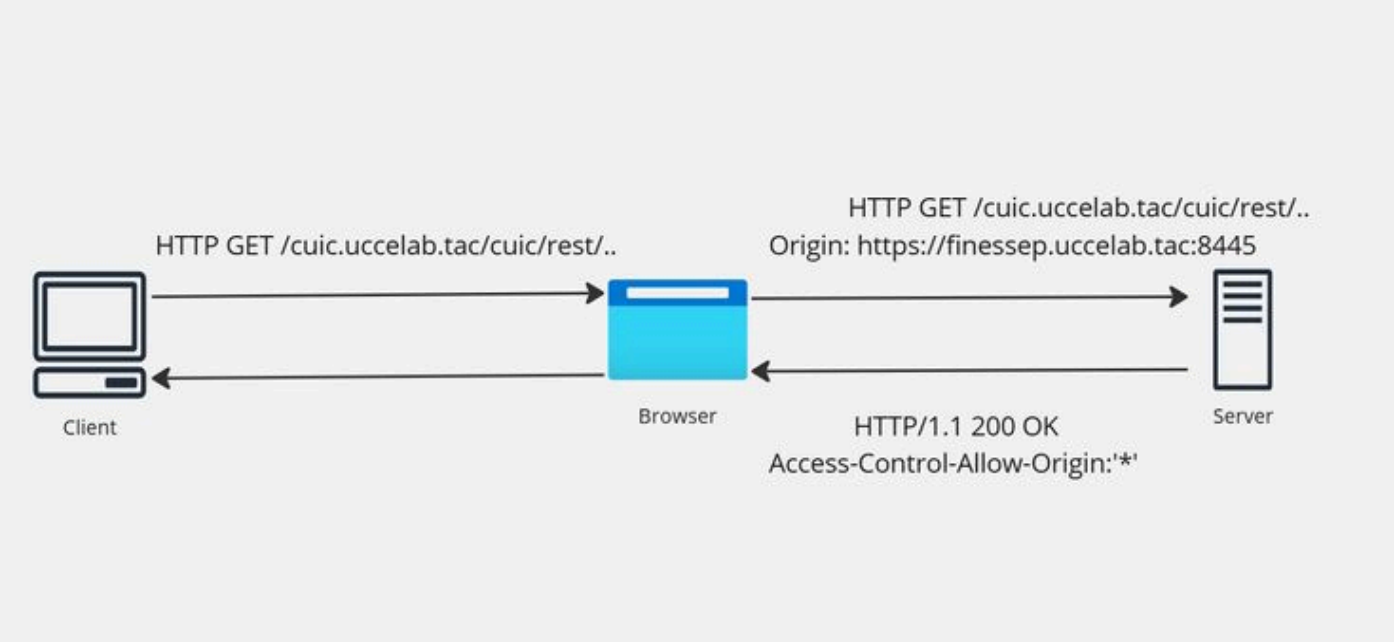
3. Il server decide come rispondere alla richiesta e invia la risposta al browser.

4. Il browser decide se il client deve avere accesso alla risposta e o passa la risposta al client o restituisce un errore.

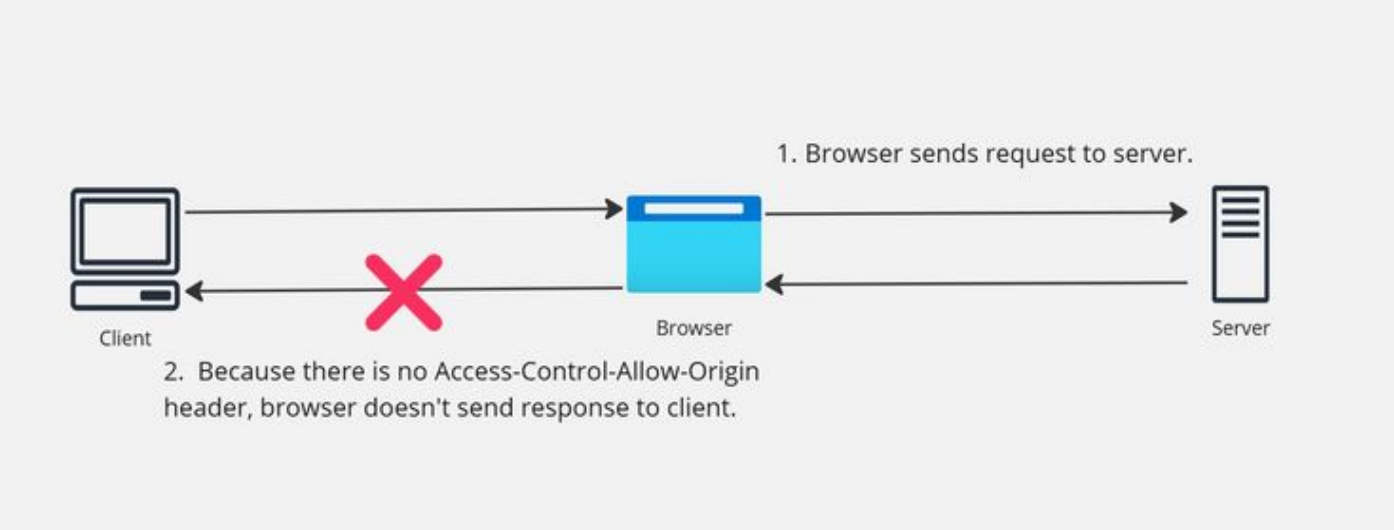


Prima di inviare una richiesta di origine incrociata, il browser aggiunge automaticamente un'intestazione di origine alla richiesta HTTP. Questa intestazione, che il client non può modificare, è una parte fondamentale di CORS e serve a identificare l'origine del client (ossia, il dominio, il protocollo e la porta da cui è stata caricata la risorsa client). Questa misura di protezione impedisce ai client di rappresentare altre origini. L'intestazione di origine è fondamentale per CORS, in quanto è il modo in cui il client indica al server da dove proviene.

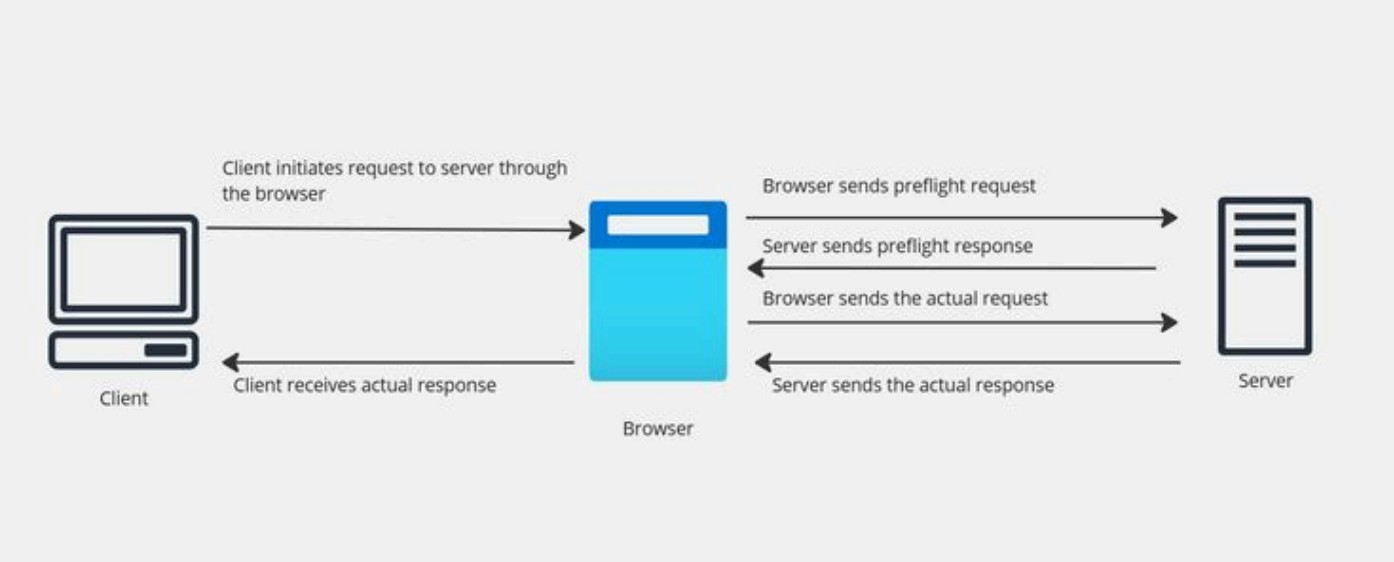
In un'interazione di condivisione risorse tra origini (CORS), l'origine del client viene identificata dall'intestazione Origine nella richiesta iniziale. Nella risposta del server viene quindi utilizzata l'intestazione Access-Control-Allow-Origin per indicare se al client è consentito accedere alla risorsa richiesta. Questa intestazione di risposta è fondamentale; se assente, la richiesta CORS non riesce. L'intestazione Access-Control-Allow-Origin può contenere un carattere jolly (*) che consente l'accesso da qualsiasi origine oppure un'origine specifica che consente l'accesso solo a un determinato client. Mentre nell'immagine è visualizzato Access-Control-Allow-Origin: *, implicando che CUIC consente tutte le origini, CUIC in genere invia questa intestazione con un'origine specifica in scenari reali.



Quando un browser rifiuta una richiesta CORS, significa che il client non riceve informazioni sulla risposta del server. Il client sa solo che si è verificato un errore, ma non dispone di dettagli sul problema specifico. Ciò può rendere difficile il debug degli errori CORS, in quanto è difficile distinguere un errore CORS da altri tipi di errori. Anche se la richiesta iniziale viene inviata al server, se la risposta del server non dispone di un'intestazione Access-Control-Allow-Origin valida, il browser blocca la risposta e attiva un errore sul lato client, impedendo al client di visualizzare la risposta dettagliata del server.



Questa immagine spiega l'intero processo CORS, con un'attenzione particolare alla fase di verifica preliminare, che è essenziale per la gestione di tipi specifici di richieste incrociate.

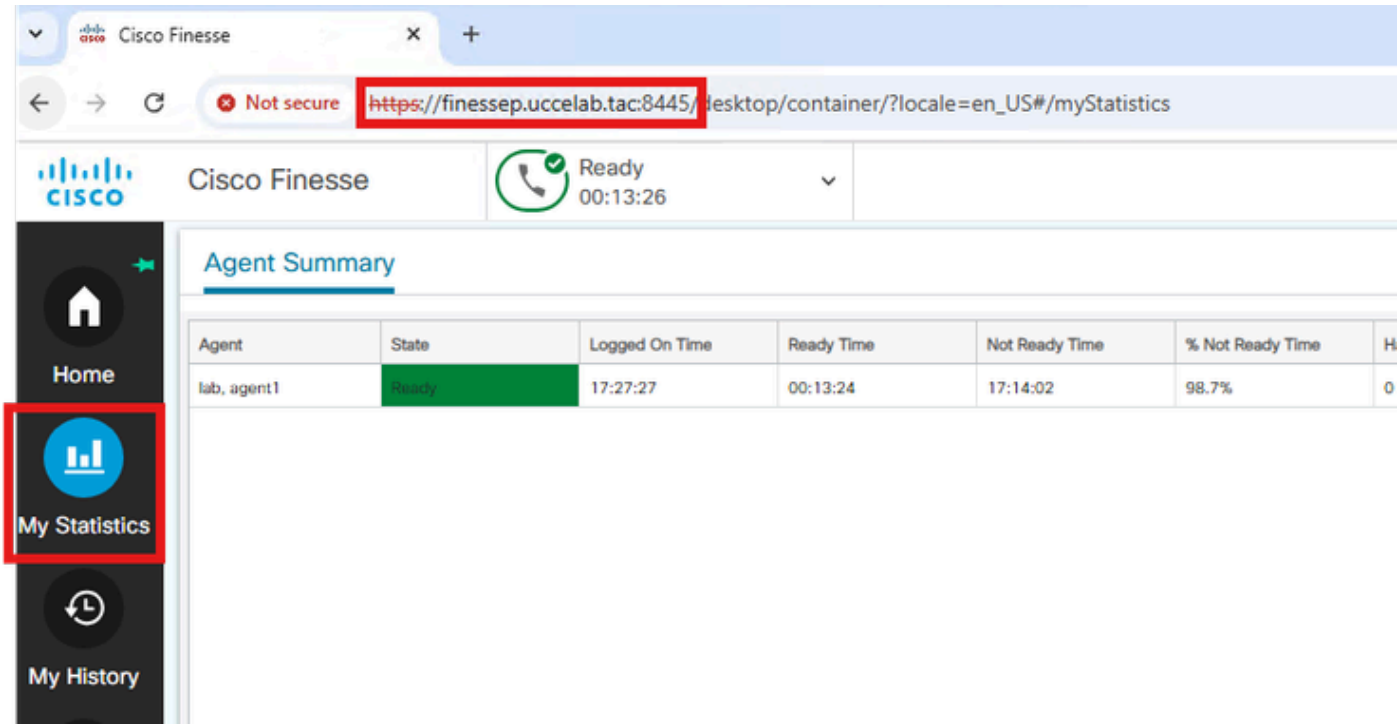


CORS in azione con Cisco Finesse

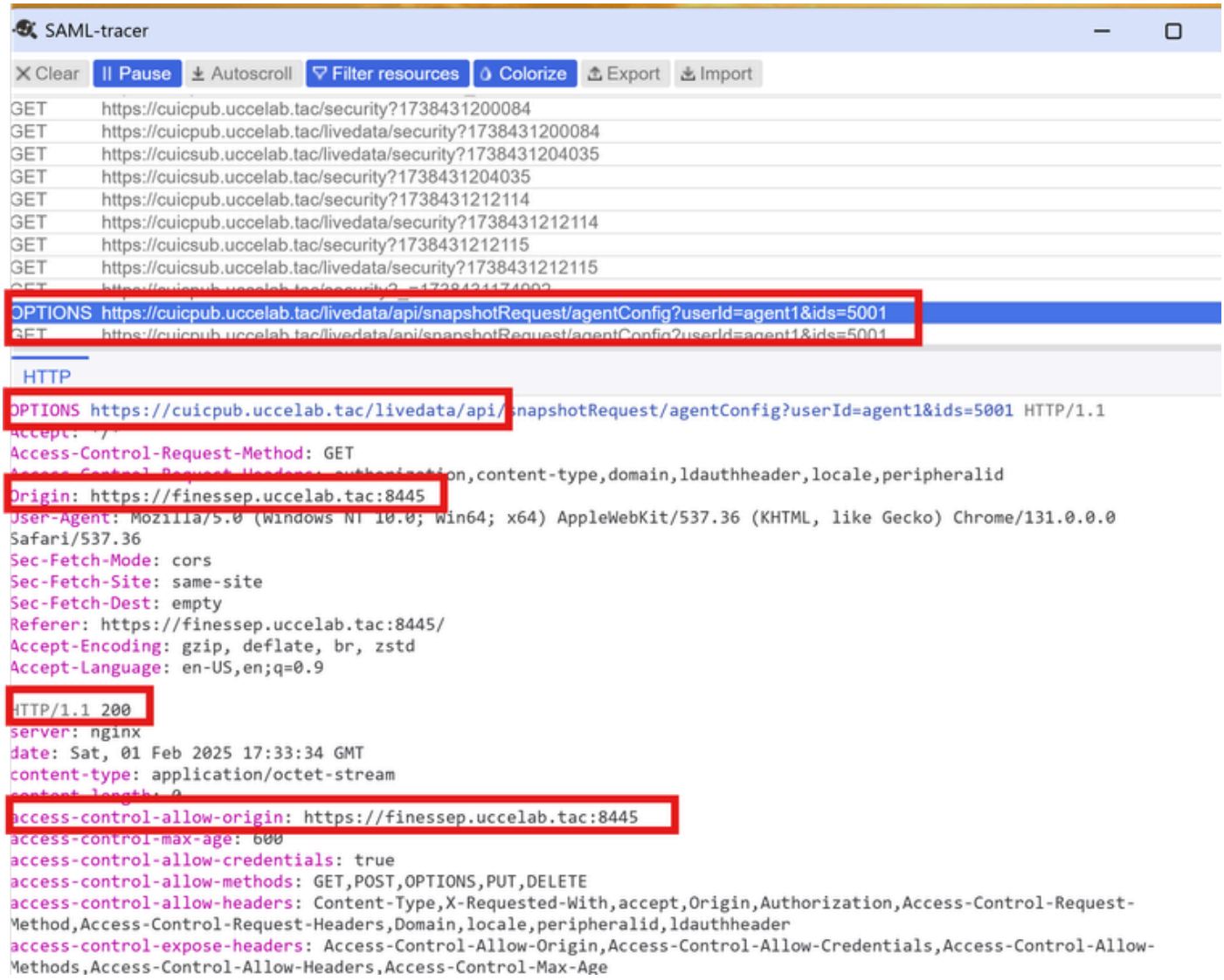
Esempio: Analisi del comportamento CORS con Live Data Gadget

In questa sezione viene descritto l'utilizzo tipico di CORS (Cross-Origin Resource Sharing) con Cisco Finesse nei contact center. Gli agenti e i supervisori in genere utilizzano Cisco Finesse per accedere ai report dei dati in tempo reale (come mostrato nell'immagine di esempio).

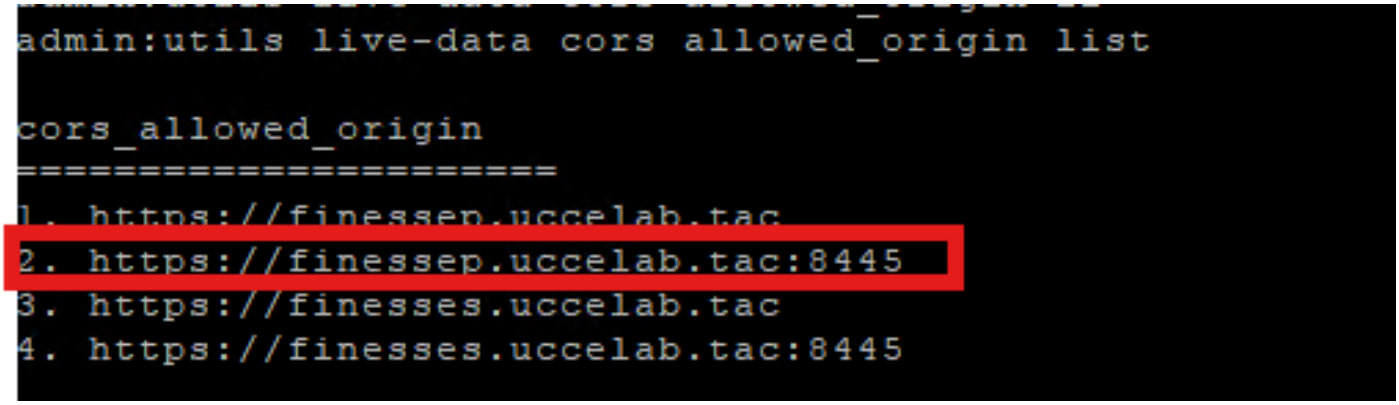
Quando un agente o un supervisore fa clic su un gadget di report, la relativa azione avvia una richiesta di recupero dei dati. Questa richiesta viene inviata dal codice JavaScript dell'applicazione Finesse (che agisce come client) al server CUIC/Live Data utilizzando un metodo GET. Come mostrato nell'immagine di tracciamento SAML, il browser invia prima una richiesta di verifica preliminare al server, il ciclo di vita CORS descritto in precedenza.

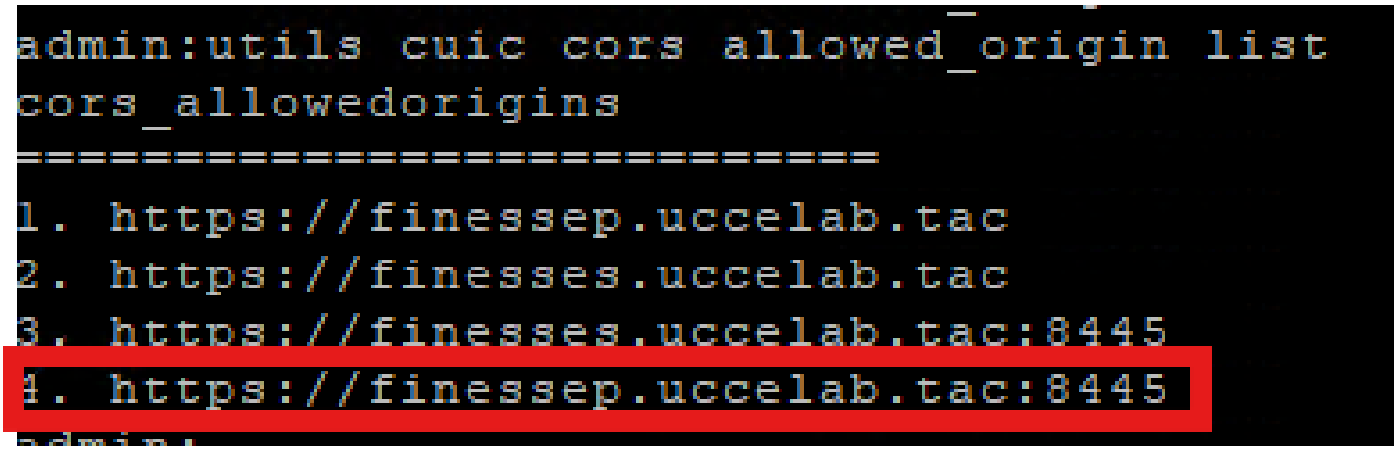


Una richiesta HTTP OPTIONS (la richiesta di verifica preliminare) viene inviata al server CUIC/Live Data. Questa richiesta specifica l'origine come nome di dominio completo (FQDN) del server Finesse, inclusa la porta 8445. Si tratta dello stesso indirizzo e della stessa porta utilizzati dagli agenti per accedere all'applicazione Cisco Finesse.



I comandi dell'interfaccia della riga di comando (CLI) sul controllo server CUIC/Live Data la cui origine è autorizzata ad accedere alle proprie risorse Live Data. Se l'origine del server Finesse (FQDN e porta) è configurata in queste impostazioni, gli agenti possono visualizzare i dettagli del gadget di dati dinamici in Finesse.





Strumento TAC per test di connessione CORS

Le configurazioni errate di CORS sul lato server possono talvolta causare problemi con gadget di dati attivi o di terze parti in Cisco Finesse. Questo articolo fornisce un collegamento a un gadget Controllo rapido CORS, uno strumento di risoluzione dei problemi progettato per aiutare a diagnosticare i problemi di Condivisione risorse tra origini che interessano i gadget Finesse, tra cui visualizzazioni di dati in tempo reale e altre integrazioni di terze parti.

Tecnicamente, questo gadget funziona inviando richieste di verifica preliminare dal client Cisco Finesse a una risorsa di destinazione specificata. Questa funzionalità di verifica rapida consente di identificare e risolvere rapidamente i problemi CORS, accelerando il processo di risoluzione dei problemi.

Per distribuire il gadget Controllo rapido CORS di Contact Center versione 12.6-v1.0 all'interno del desktop Finesse:

1. Scaricare i file dei gadget dalla cartella 12.6-v1.0 di Contact Center CORS Quick Check.2.

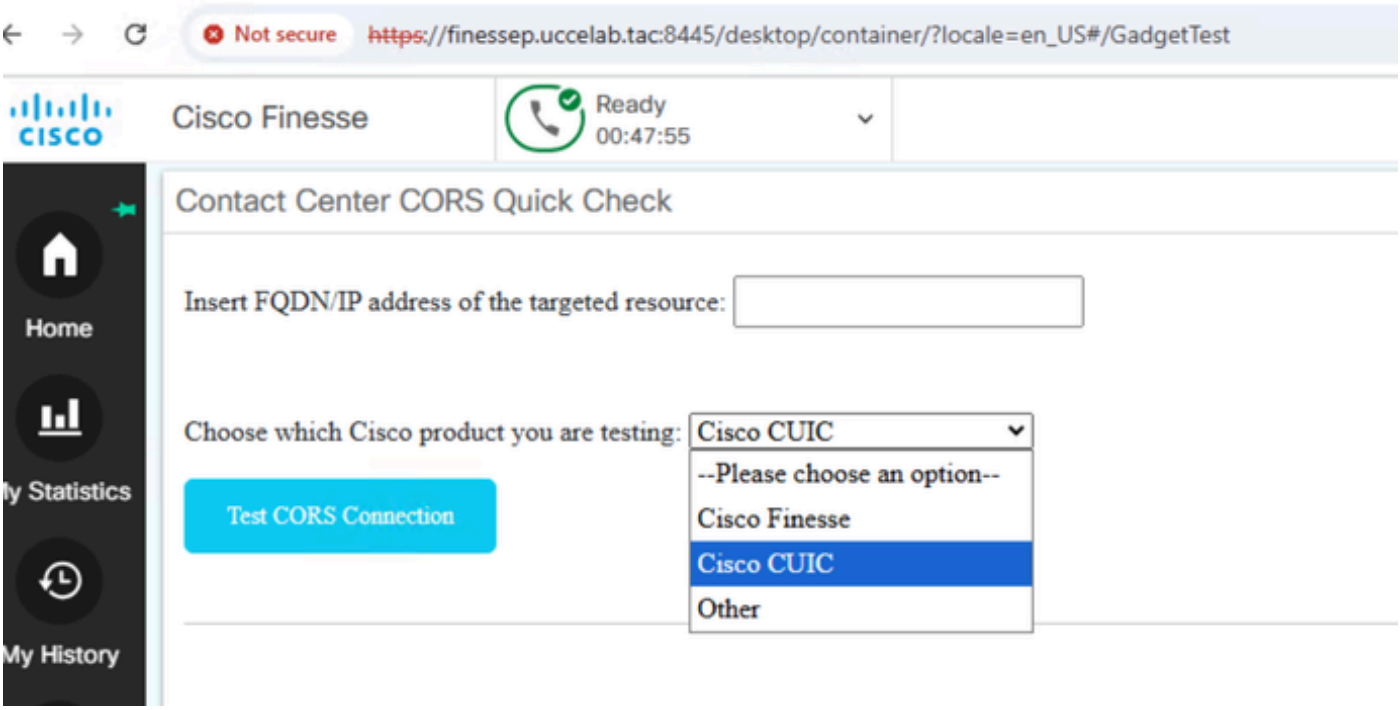
2. Copiare il contenuto della cartella Contact Center CORS Quick Check 12.6-v1.0 nella directory 3rdpartygadget all'interno dell'installazione di Finesse.

3. Aggiungere il gadget al ruolo utente desiderato (agente, supervisore e così via) nel layout del desktop Finesse. L'esempio XML fornito mostra la configurazione corretta per l'aggiunta di questo gadget.

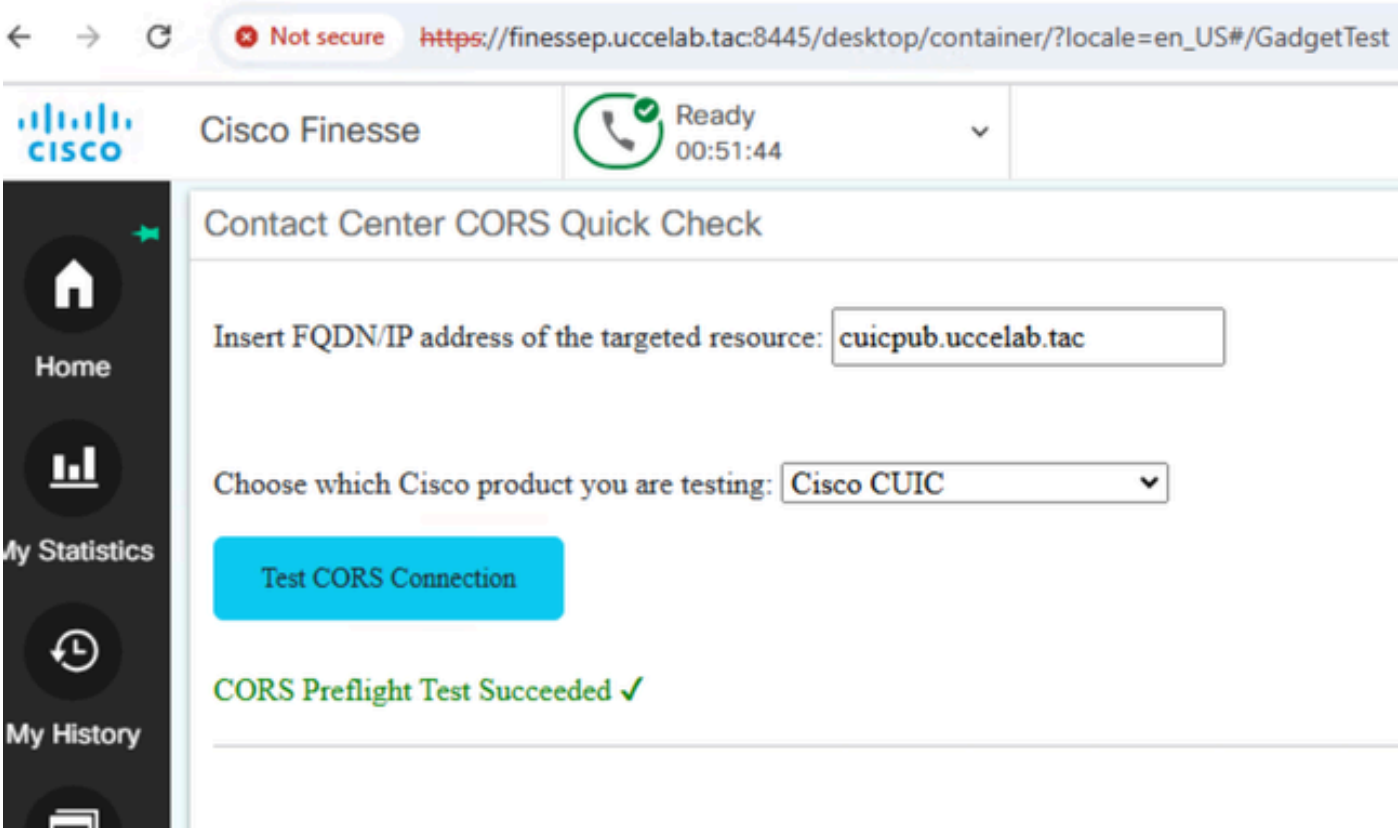
```
<gadget>/3rdpartygadget/files/TestCORSgadget.xml</gadget>
```

Vedere il capitolo relativo ai gadget di terze parti della Guida per gli sviluppatori Finesse e il capitolo relativo alla gestione dei gadget di terze parti della Guida all'amministrazione Finesse per ulteriori informazioni sul caricamento di gadget di terze parti e sulla loro aggiunta al desktop.

Dopo aver caricato i file dei gadget e riavviato il servizio Cisco Finesse Tomcat, il gadget è disponibile e visualizza l'interfaccia utente grafica (GUI).



È possibile selezionare CUIC dall'elenco a discesa in alto. Immettere il nome di dominio completo (FQDN) del server CUIC nel campo fornito. Un test di successo sarà come mostrato qui.



Se il test ha esito positivo, il server CUIC è configurato correttamente per la condivisione delle risorse tra origini (CORS) con il server Finesse. I log di traccia SAML del browser mostrano che una richiesta HTTP OPTIONS (la verifica preliminare CORS) è stata inviata al server CUIC. Questa richiesta include l'indirizzo del server Finesse nell'intestazione Origin. Il server CUIC ha risposto con un messaggio HTTP 200 OK e, cosa importante, l'intestazione Access-Control-Allow-

Origin nella risposta contiene anche l'indirizzo del server Finesse. Ciò conferma che il server CUIC è configurato per consentire le richieste provenienti dall'origine del server Finesse, verificando che CORS sia impostato correttamente.

<#root>

```
OPTIONS https://cuicpub.uccelab.tac/cuic/ HTTP/1.1

sec-ch-ua-platform: "Windows"
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome..
sec-ch-ua: "Google Chrome";v="131", "Chromium";v="131", "Not_A Brand";v="24"
sec-ch-ua-mobile: ?0
Accept: */*

Origin: https://finessep.uccelab.tac:8445

Sec-Fetch-Site: same-site
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: https://finessep.uccelab.tac:8445/
Accept-Encoding: gzip, deflate, br, zstd
Accept-Language: en-US,en;q=0.9
```

<#root>

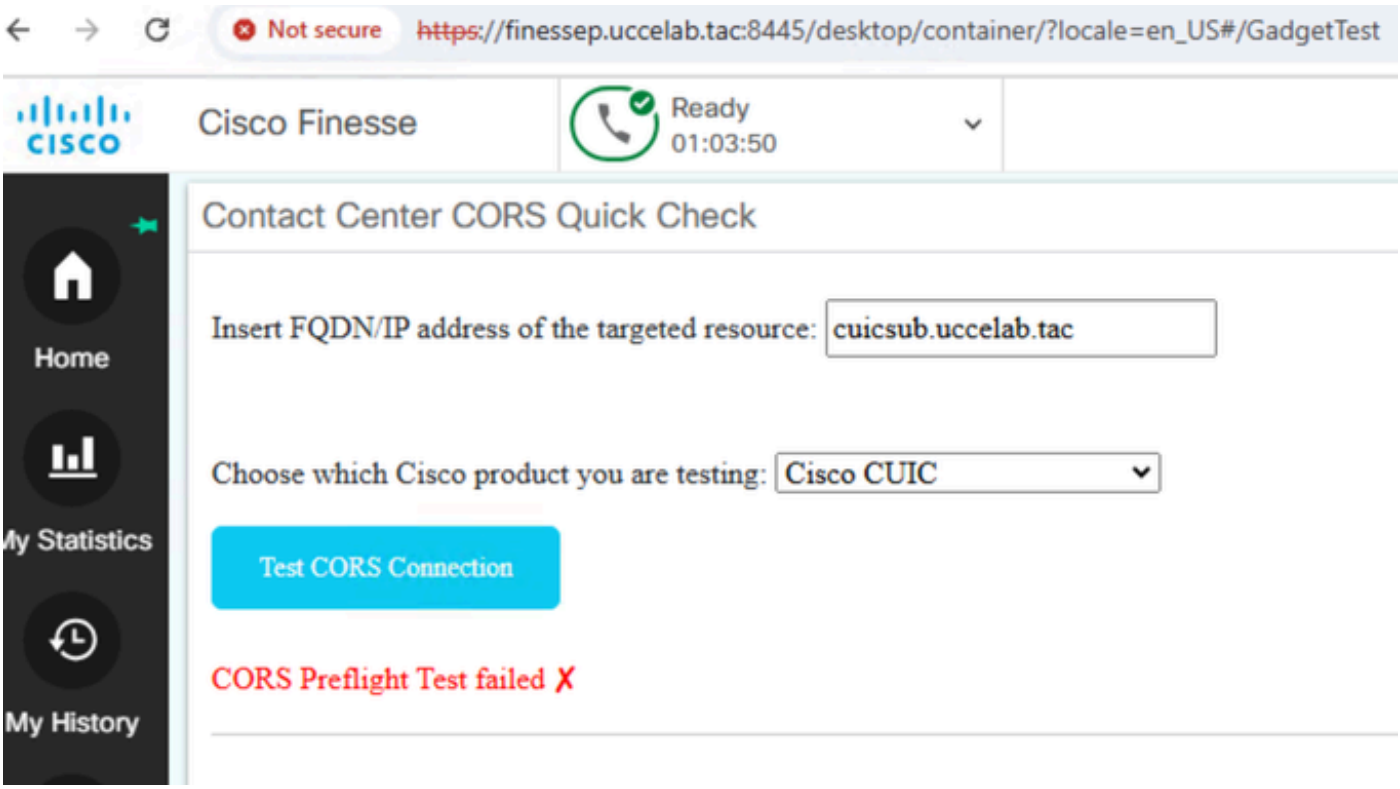
```
HTTP/1.1 200

server: nginx
date: Sat, 08 Feb 2025 01:27:47 GMT
content-length: 0
strict-transport-security: max-age=31536000; includeSubDomains
set-cookie: JSESSIONID=bE73993C4A7C1Fc1b33A7AaF897B8428; Path=/cuic; Secure; HttpOnly; SameSite=Strict
pragma: No-cache
cache-control: no-cache
expires: Thu, 01 Jan 1970 00:00:00 GMT
x-frame-options: SAMEORIGIN
x-xss-protection: 1; mode=block
x-content-type-options: nosniff
content-security-policy: default-src 'self' ; script-src 'self' data: 'unsafe-inline' 'unsafe-eval' ; s
vary: origin,access-control-request-method,Access-Control-Request-Headers

access-control-allow-origin: https://finessep.uccelab.tac:8445

access-control-allow-credentials: true
access-control-expose-headers: access-control-allow-origin,access-control-allow-credentials,access-cont
access-control-max-age: 600
access-control-allow-methods: DELETE,POST,GET,OPTIONS,PUT
access-control-allow-headers: referer,peripheralid,origin,access-control-request-method,locale,accept,a
allow: GET,POST,OPTIONS,PUT,DELETE
```

In questo scenario, lo strumento mostra una configurazione non funzionante. A differenza dell'esempio precedente, il server Finesse non è configurato come sottoscrittore sul server CUIC. ma è configurato solo sull'editore CUIC. Di conseguenza, la richiesta di verifica preliminare CORS ha esito negativo e il server CUIC risponde con un errore HTTP 403 (accesso negato).



<#root>

OPTIONS https://cuicsub.uccelab.tac/cuic/ HTTP/1.1

Accept: */*
Access-Control-Request-Method: OPTIONS

Origin: https://finessep.uccelab.tac:8445

User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome..
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-site
Sec-Fetch-Dest: empty
Referer: https://finessep.uccelab.tac:8445/
Accept-Encoding: gzip, deflate, br, zstd
Accept-Language: en-US,en;q=0.9

<#root>

HTTP/1.1 403

server: nginx
date: Sat, 08 Feb 2025 01:54:52 GMT
content-type: text/html;charset=utf-8
content-length: 2143
strict-transport-security: max-age=31536000; includeSubDomains
set-cookie: JSESSIONID=1C7606841B83d7847486c3d18D31cEfD; Path=/cuic; Secure; HttpOnly; SameSite=Strict
pragma: No-cache
cache-control: no-cache
expires: Thu, 01 Jan 1970 00:00:00 GMT
x-frame-options: SAMEORIGIN
x-xss-protection: 1; mode=block
x-content-type-options: nosniff

Come mostrato dall'output dell'interfaccia della riga di comando (CLI) del sottoscrittore CUIC, Cisco Finesse non è presente nell'elenco. Ciò indica che Finesse non è attualmente configurato come sottoscrittore in questo server CUIC.

<#root>

```
admin:utils cuic cors allowed_origin list

cors_allowedorigins
==========================
1. https://finessep.uccelab.tac
2. https://finesses.uccelab.tac
3. https://finesses.uccelab.tac:8445
```

Informazioni su questa traduzione

Cisco ha tradotto questo documento utilizzando una combinazione di tecnologie automatiche e umane per offrire ai nostri utenti in tutto il mondo contenuti di supporto nella propria lingua. Si noti che anche la migliore traduzione automatica non sarà mai accurata come quella fornita da un traduttore professionista. Cisco Systems, Inc. non si assume alcuna responsabilità per l’accuratezza di queste traduzioni e consiglia di consultare sempre il documento originale in inglese (disponibile al link fornito).