Comprender el uso compartido de recursos entre orígenes (CORS) para Finesse

Contenido

Introducción

Este documento describe completamente el uso compartido de recursos entre orígenes para que, durante la resolución de problemas, los procesos subyacentes se entiendan a fondo.

Prerequisites

Requirements

Cisco recomienda que tenga conocimiento sobre estos temas:

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

Componentes Utilizados

La información que contiene este documento se basa en las siguientes versiones de software y hardware.

- UCCE versión 12.6.2
- Finesse versión 12.6.2
- CUIC versión 12.6.2

La información que contiene este documento se creó a partir de los dispositivos en un ambiente de laboratorio específico. Todos los dispositivos que se utilizan en este documento se pusieron en

funcionamiento con una configuración verificada (predeterminada). Si tiene una red en vivo, asegúrese de entender el posible impacto de cualquier comando.

Antecedentes

Qué es CORS

El uso compartido de recursos entre orígenes (CORS) es una forma en que los servidores pueden controlar qué sitios web (dominios, protocolos y puertos) tienen permiso para acceder a sus recursos. Mientras que los navegadores normalmente bloquean las solicitudes de diferentes orígenes (la política del mismo origen), CORS da a los servidores el poder de relajar selectivamente esta restricción. Básicamente, los servidores utilizan encabezados HTTP especiales para indicar al explorador qué orígenes están permitidos, qué tipos de solicitudes están permitidos (como GET, POST, etc.) y qué encabezados personalizados se pueden incluir. Esto permite a los servidores decidir quién puede acceder a sus API y cómo, desde acceso completamente abierto hasta acceso estrictamente limitado. CORS funciona haciendo que el navegador y el servidor se comuniquen a través de estos encabezados HTTP para administrar las solicitudes de origen cruzado.

CORS utiliza encabezados HTTP para habilitar las solicitudes de origen cruzado controladas. El explorador y el servidor se comunican a través de estos encabezados, y el servidor especifica los orígenes, métodos y encabezados permitidos. Si faltan los encabezados de respuesta del servidor o estos no son válidos, el explorador bloquea la respuesta y aplica la directiva del mismo origen. Para ciertas solicitudes, el navegador primero envía una solicitud de comprobación previa al servidor para asegurarse de que acepta la solicitud real de origen cruzado.

Los exploradores utilizan solicitudes de comprobación previa para comprobar si un servidor permite una solicitud de origen cruzado antes de enviar la solicitud real. Estas solicitudes de comprobaciones incluyen detalles como el método HTTP y los encabezados personalizados. Los servidores habilitados para CORS pueden entonces responder, permitiendo o denegando la solicitud real. Si un servidor no está configurado para CORS, no responde correctamente a la comprobación previa y el navegador bloquea la solicitud real, protegiendo así al servidor del acceso cruzado no deseado.

El uso compartido de recursos entre orígenes (CORS) es crucial para la seguridad y funcionalidad web. Permite el acceso controlado a recursos de diferentes orígenes (dominios, protocolos, puertos), lo que es necesario porque los exploradores aplican una directiva del mismo origen que normalmente bloquea dicho acceso.

Ciclo de vida de un CORS

Una solicitud CORS consta de dos partes: el cliente que realiza la solicitud y el servidor que recibe la solicitud. En el lado del cliente, el desarrollador escribe código JavaScript para enviar la solicitud al servidor. El servidor responde a la solicitud estableciendo encabezados específicos de CORS especiales para indicar que se permite la solicitud de origen cruzado. Sin la participación del cliente y del servidor, la solicitud CORS falla.

Los actores clave en una solicitud CORS son el cliente, el navegador y el servidor. El cliente desea algún dato del servidor, como una respuesta de la API JSON o el contenido de una página web. El explorador actúa como intermediario de confianza para comprobar que el cliente puede tener acceso a los datos desde el servidor.

Cliente:

El cliente es un fragmento de código JavaScript que se ejecuta en un sitio web y es responsable de iniciar la solicitud CORS

Nota: Finesse es una aplicación web. Se instala en un servidor, y los agentes acceden a él simplemente utilizando sus navegadores web, eliminando la necesidad de instalaciones en el lado del cliente o mantenimiento de plugins u otro software. Como se ha demostrado en el ejemplo de CORS en acción con Cisco Finesse, esta arquitectura admite funciones como informes de datos en directo. En este contexto, el código JavaScript del gadget de datos activos de Cisco Finesse actúa como cliente, mientras que Cisco CUIC actúa como servidor dentro del ciclo de vida de CORS. Básicamente, el cliente Finesse basado en navegador interactúa con el servidor CUIC para recuperar datos activos.

Cliente frente a usuario:

A veces las palabras cliente y usuario se utilizan indistintamente, pero son diferentes en el contexto de CORS. Un usuario es una persona que visita un sitio web o un usuario de Finesse (agente o supervisor) que accede a Finesse en este contexto, mientras que un cliente es el código real servido por ese sitio web. Varios usuarios pueden visitar el mismo sitio web y recibir el mismo código de cliente JavaScript.

Explorador:

El explorador, también conocido como agente de usuario, aloja el código de cliente. Desempeña un papel fundamental en el CORS al añadir información adicional a las solicitudes salientes, permitiendo al servidor identificar al cliente. Además, el navegador interpreta la respuesta del servidor, determinando si entregar los datos al cliente o devolver un error. Estas acciones en el navegador son esenciales para mantener la seguridad proporcionada por la política del mismo origen. Sin la aplicación por parte del navegador de las reglas CORS, los clientes podrían realizar solicitudes no autorizadas, poniendo en peligro este mecanismo de seguridad vital.

Servidor:

El servidor es el destino de la solicitud CORS y es el ejemplo de gadget de CUIC para datos en directo con Cisco Finesse. El servidor almacena los datos que el cliente desea y tiene la última palabra sobre si la solicitud CORS está permitida o no.

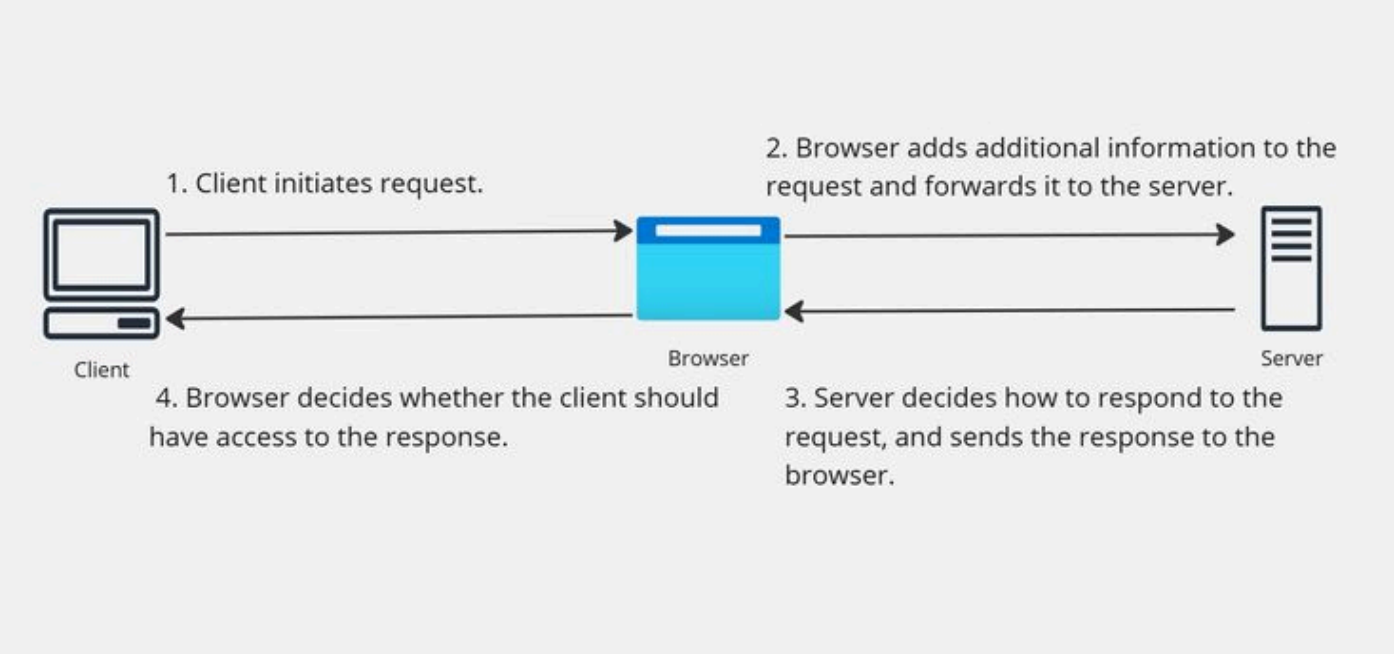
Ahora que sabe quién participa en una solicitud CORS, echemos un vistazo a cómo funcionan todos juntos. Las imágenes siguientes ilustran el ciclo de vida del CORS de alto nivel:

1. El cliente inicia la solicitud.

2. El explorador agrega información adicional a la solicitud y la reenvía al servidor.

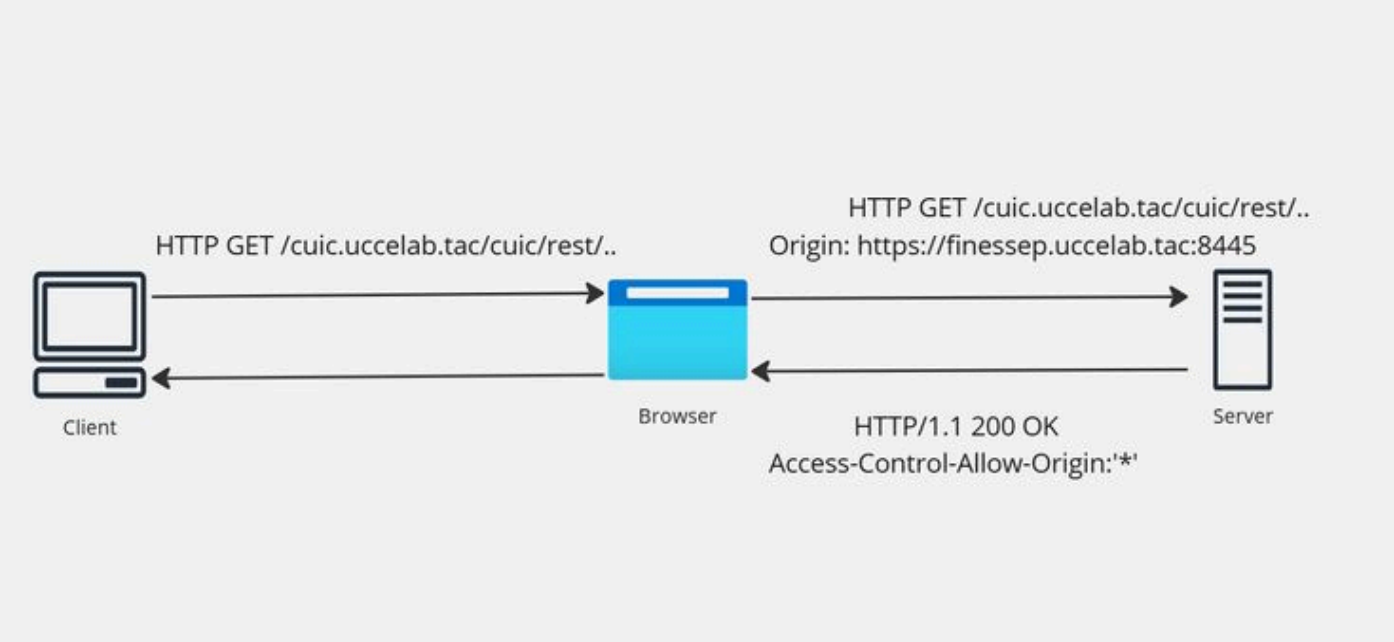
3. El servidor decide cómo responder a la solicitud y envía la respuesta al navegador.

4. El navegador decide si el cliente debe tener acceso a la respuesta y, o bien pasa la respuesta al cliente o devuelve un error.

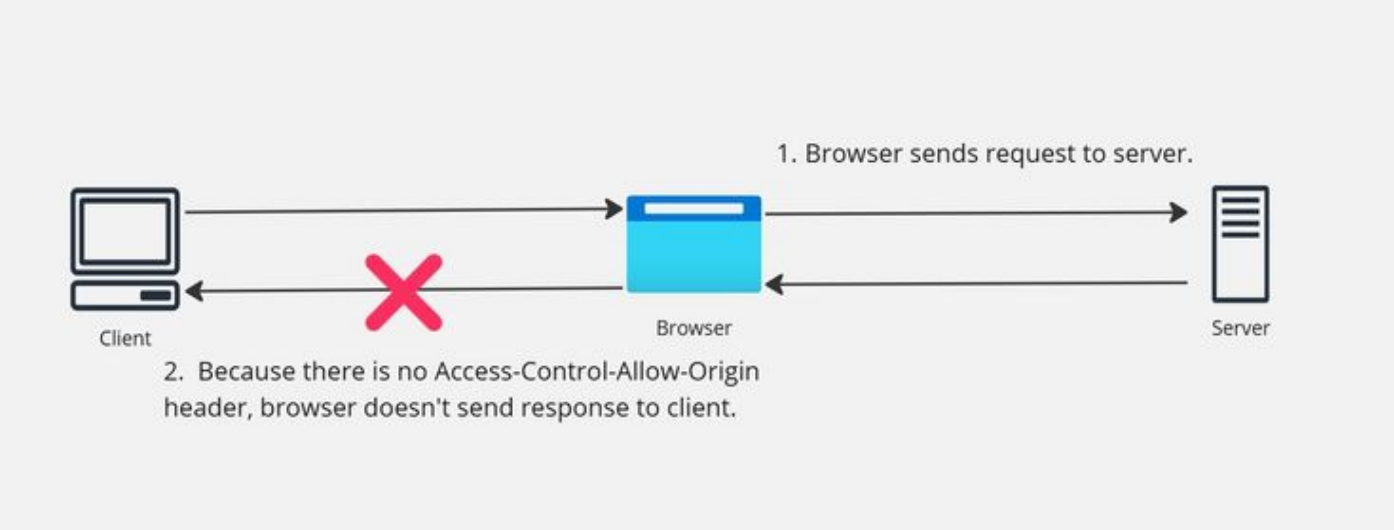


Antes de enviar una solicitud de origen cruzado, el explorador agrega automáticamente un encabezado de origen a la solicitud HTTP. Este encabezado, que el cliente no puede modificar, es una parte crucial de CORS y sirve para identificar el origen del cliente (es decir, el dominio, el protocolo y el puerto desde el cual se cargó el recurso del cliente). Esta medida de seguridad impide que los clientes suplanten otros orígenes. El encabezado de origen es fundamental para CORS, ya que es la forma en que el cliente le dice al servidor de dónde viene.

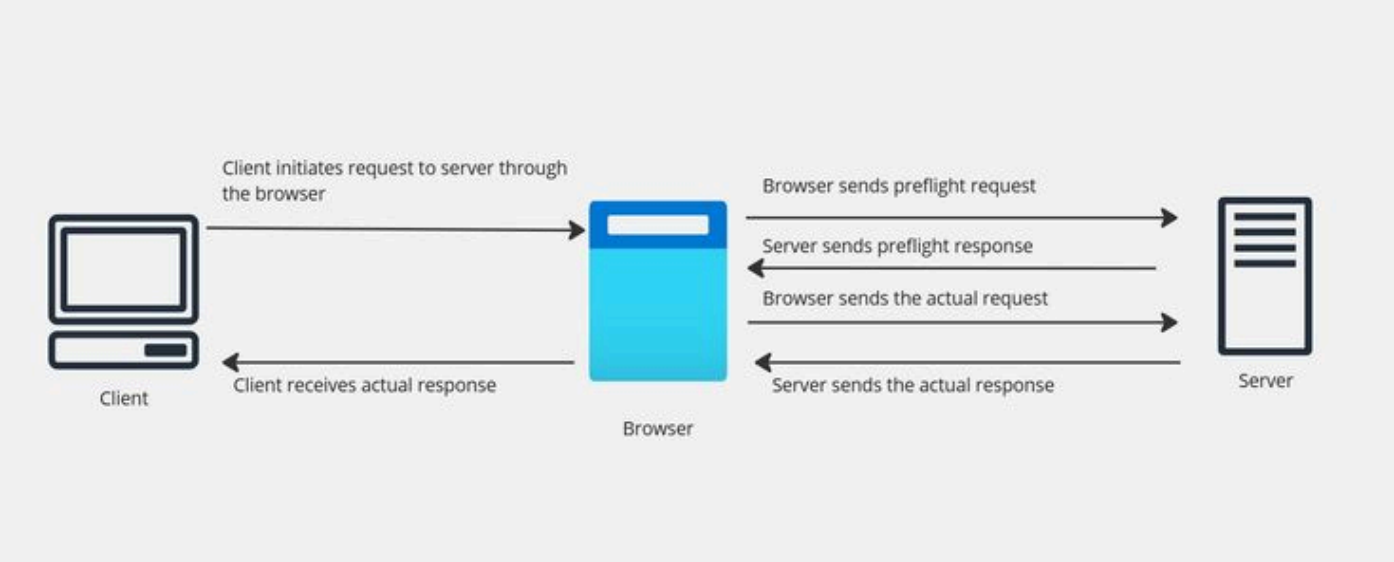
En una interacción de uso compartido de recursos entre orígenes cruzados (CORS), el origen del cliente se identifica mediante el encabezado Origin en la solicitud inicial. A continuación, el servidor utiliza el encabezado Access-Control-Allow-Origin en su respuesta para indicar si el cliente tiene permiso para obtener acceso al recurso solicitado. Este encabezado de respuesta es crucial; si no existe, la solicitud CORS falla. El encabezado Access-Control-Allow-Origin puede contener un comodín (*), permitiendo el acceso desde cualquier origen, o un origen específico, otorgando acceso sólo a ese cliente en particular. Mientras que la imagen muestra Access-Control-Allow-Origin: *, lo que implica que CUIC permite todos los orígenes, CUIC normalmente envía este encabezado con un origen específico en escenarios reales.



Cuando un navegador rechaza una solicitud CORS, significa que el cliente no recibe información sobre la respuesta del servidor. El cliente solo sabe que se ha producido un error, pero carece de detalles sobre el problema específico. Esto puede hacer que la depuración de errores CORS sea un desafío, ya que es difícil distinguir una falla CORS de otros tipos de errores. Aunque la solicitud inicial se envía al servidor, si la respuesta del servidor carece de un encabezado Access-Control-Allow-Origin válido, el explorador bloquea la respuesta y activa un error en el cliente, lo que impide que el cliente vea la respuesta detallada del servidor.



Esta imagen explica todo el proceso CORS, con un enfoque particular en la etapa de comprobación previa, que es esencial para manejar tipos específicos de solicitudes de origen cruzado.

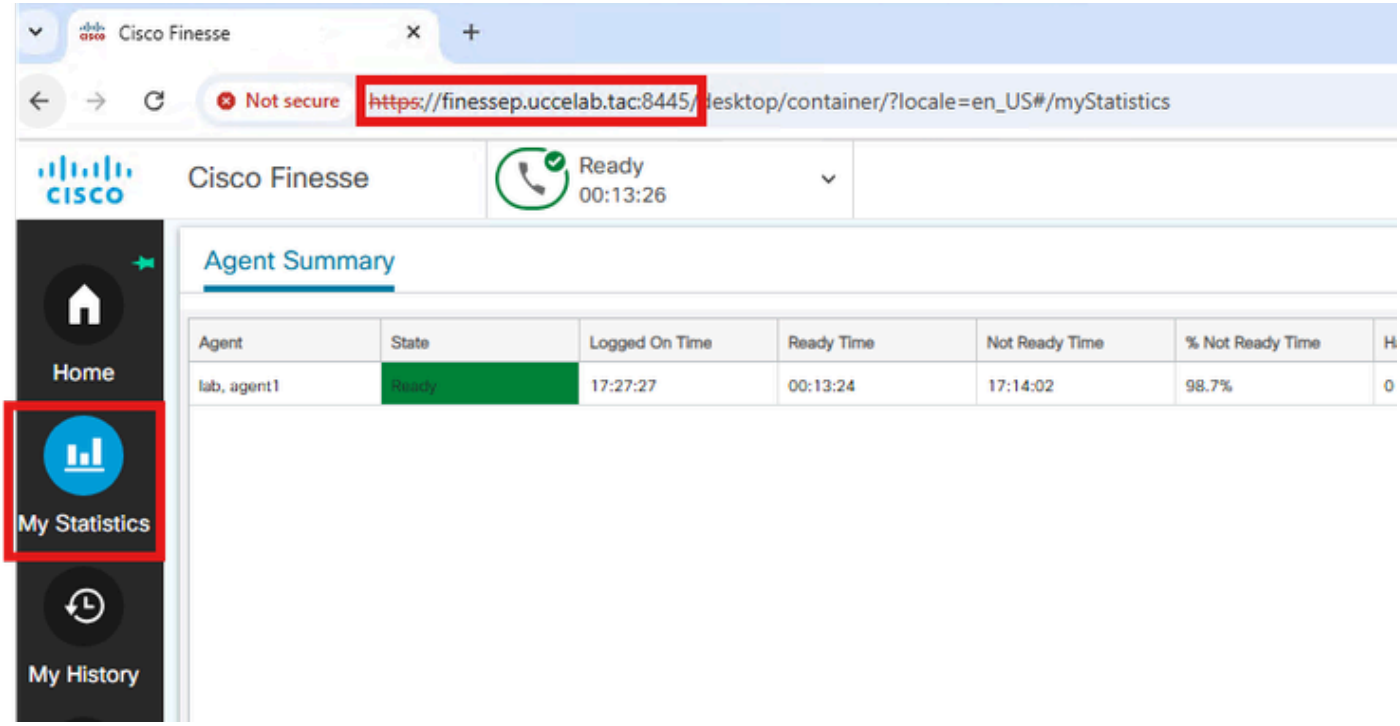


CORS en acción con Cisco Finesse

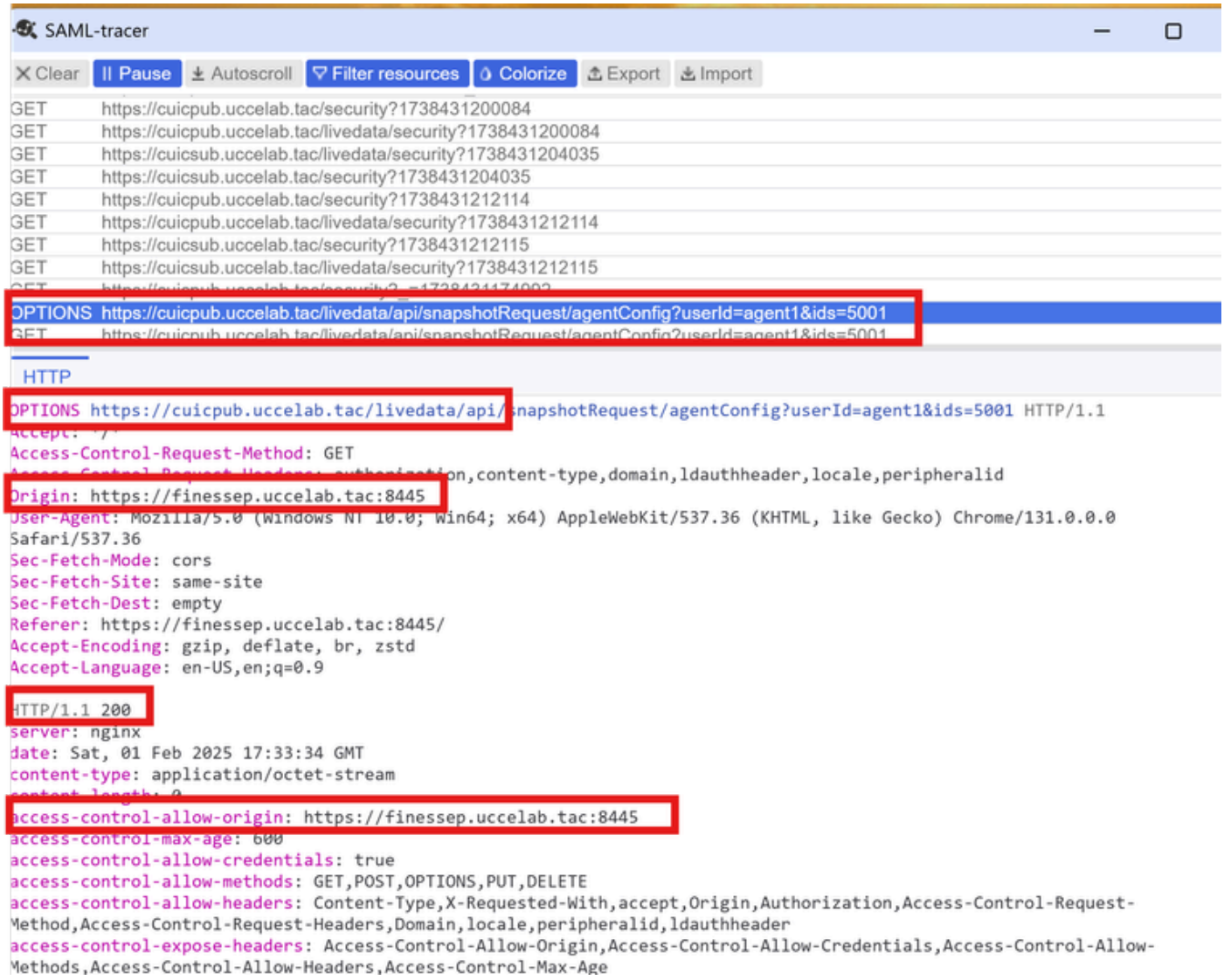
Ejemplo ilustrativo: Análisis del comportamiento del CORS con el gadget Datos en directo

En esta sección se describe el uso típico de Cross-Origin Resource Sharing (CORS) con Cisco Finesse en los Contact Centers. Los agentes y supervisores suelen utilizar Cisco Finesse para acceder a los informes de datos en tiempo real (como se muestra en la imagen de ejemplo).

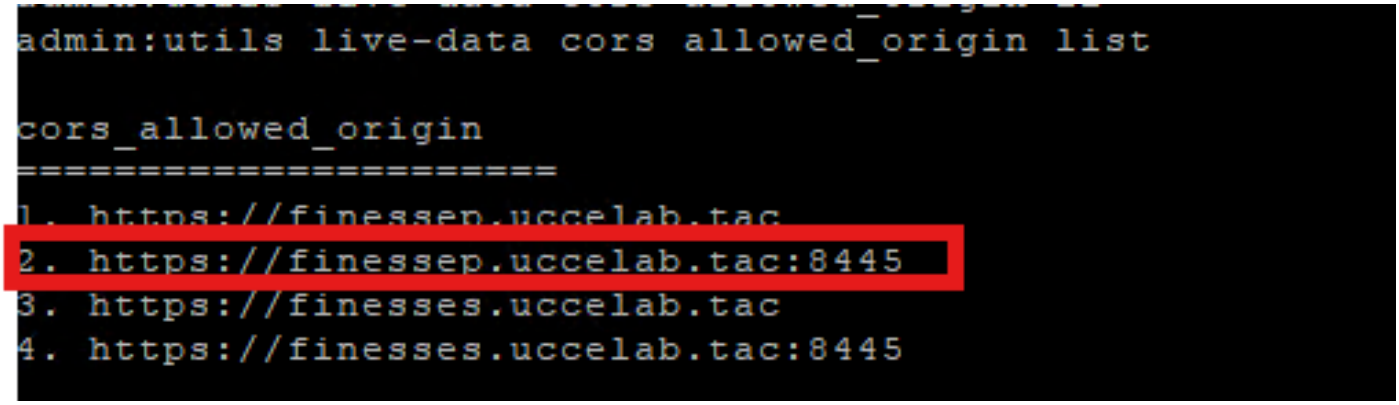
Cuando un agente o supervisor hace clic en un gadget de informes, su acción inicia una solicitud de recuperación de datos. Esta solicitud se envía desde el código JavaScript de la aplicación Finesse (que actúa como cliente) al servidor de CUIC/Live Data mediante un método GET. Como se muestra en la imagen del rastreador SAML, el navegador primero envía una solicitud de comprobación previa al servidor, el ciclo de vida CORS descrito anteriormente.

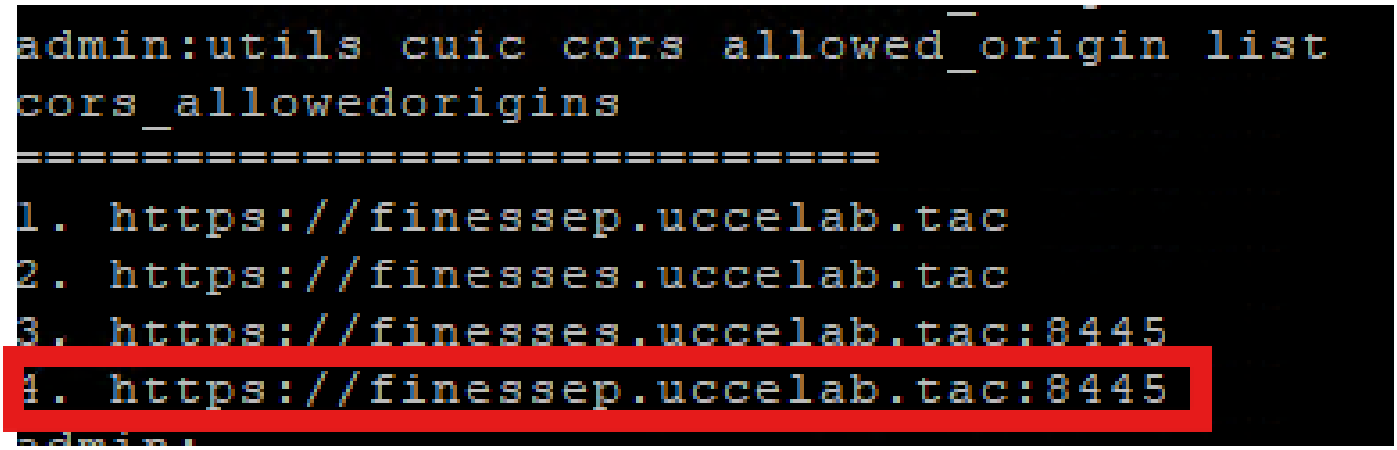


Se envía una solicitud de opciones HTTP (la solicitud de comprobación preliminar) al servidor de CUIC/Live Data. Esta solicitud especifica el origen como el nombre de dominio completo (FQDN) del servidor Finesse, incluido el puerto 8445. Se trata de la misma dirección y puerto que utilizan los agentes para acceder a la aplicación Cisco Finesse.



Los comandos de la interfaz de línea de comandos (CLI) del control de servidor CUIC/Live Data cuyo origen está permitido para tener acceso a sus recursos de datos activos. Si el origen del servidor Finesse (su FQDN y puerto) se configura en estos parámetros, los agentes podrán ver los detalles del gadget de datos en directo en Finesse.





Herramienta TAC para pruebas de conexión CORS

Los errores de configuración de CORS en el servidor pueden causar problemas con gadgets de datos activos o de terceros en Cisco Finesse. En este artículo se proporciona un enlace a un gadget de comprobación rápida de CORS. Se ha diseñado una herramienta de solución de problemas para ayudar a diagnosticar los problemas de uso compartido de recursos entre orígenes que afectan a los gadgets de Finesse, incluidas las visualizaciones de datos en directo y otras integraciones de terceros.

Técnicamente, este gadget funciona enviando solicitudes de comprobaciones desde el cliente Cisco Finesse a un recurso de destino especificado. Esta función de comprobación rápida ayuda a identificar y resolver rápidamente los problemas relacionados con CORS, lo que acelera el proceso de solución de problemas.

Para implementar el gadget Comprobación rápida 12.6-v1.0 de CORS del Contact Center en el escritorio de Finesse:

1. Descargue los archivos del gadget desde Contact Center CORS Quick Check 12.6-v1.0 folder.2.

2. Copie el contenido de la carpeta Contact Center CORS Quick Check 12.6-v1.0 en el directorio 3rdpartygadget de la instalación de Finesse.

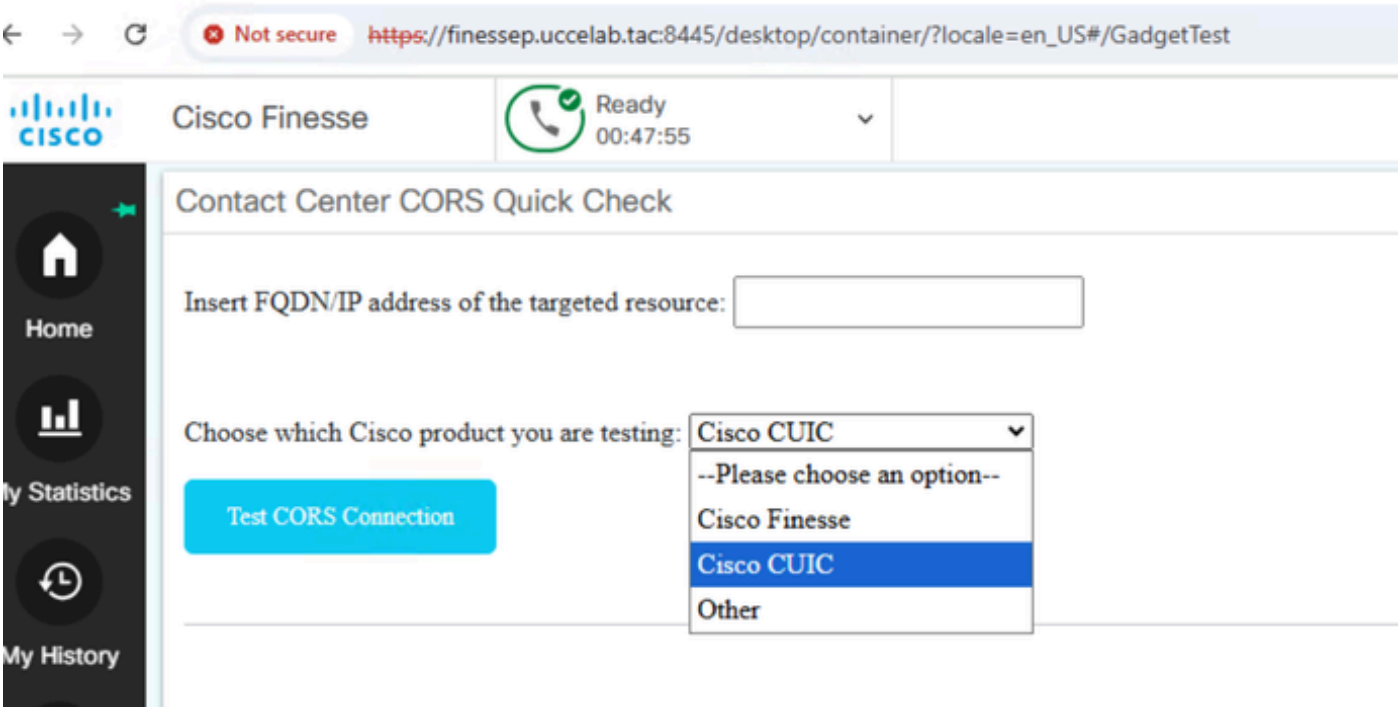
3. Agregue el gadget a la función de usuario deseada (agente, supervisor, etc.) en el diseño de escritorio de Finesse. El XML de ejemplo proporcionado muestra la configuración correcta para agregar este gadget.

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

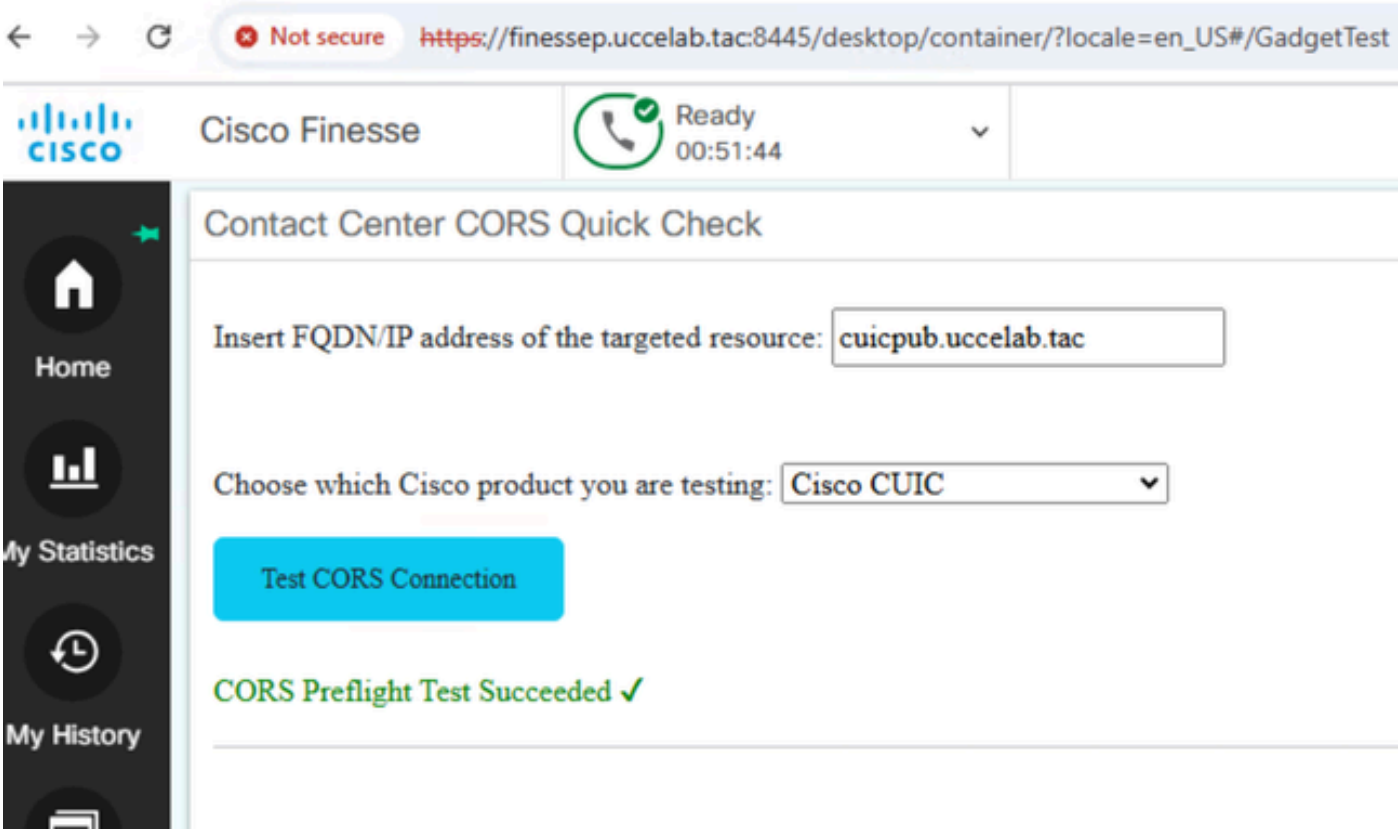
Consulte el capítulo Gadgets de terceros en la guía del desarrollador de Finesse y el capítulo Gestionar gadgets de terceros en la guía de administración de Finesse para obtener más información sobre cómo cargar gadgets de terceros y agregarlos al escritorio.

Una vez cargados los archivos del gadget y reiniciado el servicio Tomcat de Cisco Finesse, el

gadget estará disponible y mostrará la interfaz gráfica de usuario (GUI).



Puede seleccionar CUIC en la lista desplegable superior de la parte superior. Introduzca el nombre de dominio completo (FQDN) del servidor de CUIC en el campo proporcionado. Una prueba exitosa va a ser como se muestra aquí.



Una prueba correcta significa que el servidor CUIC está correctamente configurado para el uso compartido de recursos entre orígenes (CORS) con el servidor Finesse. Los registros del rastreador SAML del explorador muestran que se envió una solicitud de opciones HTTP

(comprobación previa CORS) al servidor CUIC. Esta solicitud incluía la dirección del servidor Finesse en el encabezado Origin. El servidor CUIC respondió con un mensaje HTTP 200 OK y, lo que es más importante, el encabezado Access-Control-Allow-Origin de la respuesta también contenía la dirección del servidor Finesse. Esto confirma que el servidor CUIC está configurado para permitir solicitudes del origen del servidor Finesse, verificando que CORS está configurado correctamente.

<#root>

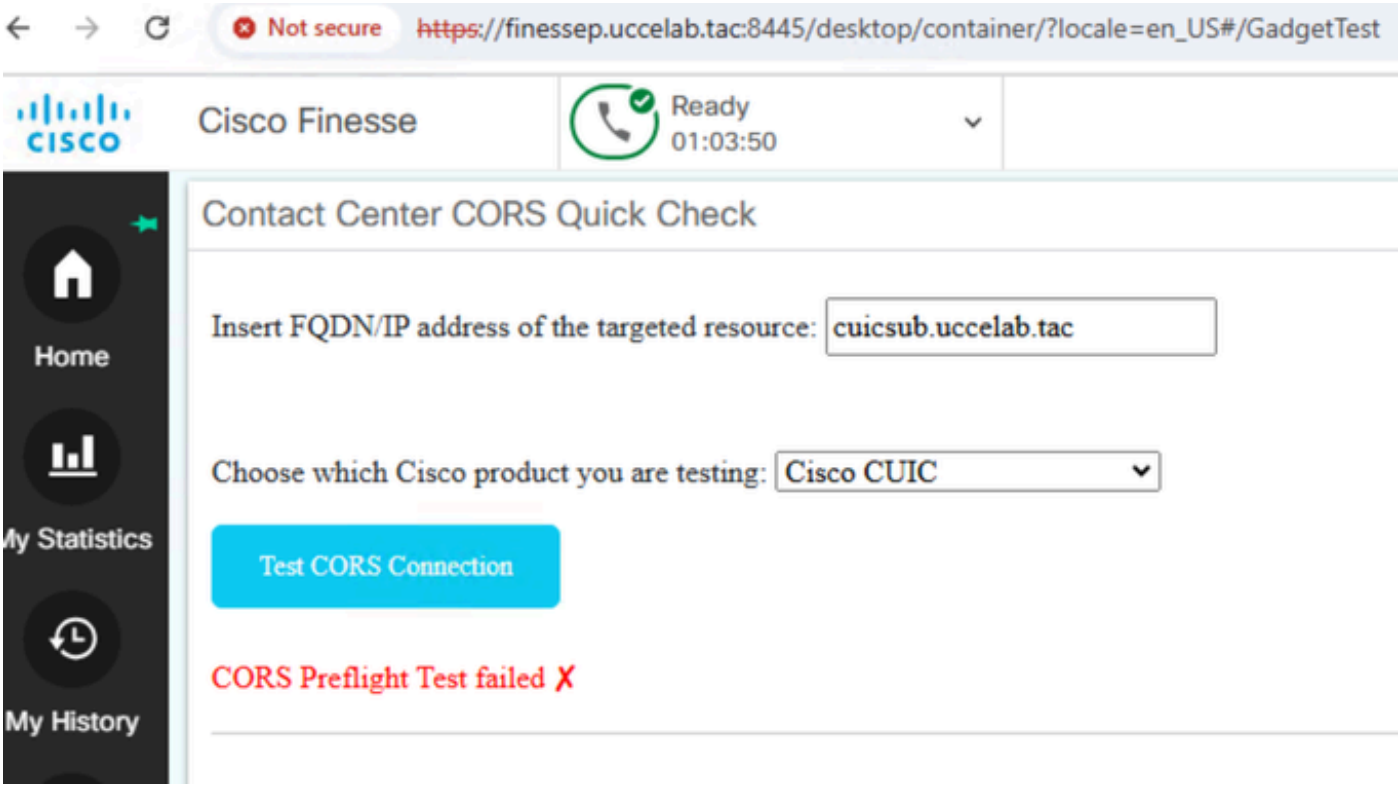
```
OPTIONS https://cuicpub.uccelab.tac/cuic/ HTTP/1.1

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
```

En este escenario, la herramienta muestra una configuración que no funciona. A diferencia del

ejemplo anterior, el servidor Finesse no está configurado como suscriptor en el servidor CUIC. En su lugar, sólo se configura en el editor de CUIC. Como resultado, la solicitud de comprobación previa de CORS falla y el servidor CUIC responde con un error HTTP 403 (Prohibido).



<#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
```

Como puede ver en la salida de la interfaz de línea de comandos (CLI) del suscriptor de CUIC, Cisco Finesse no aparece en la lista. Esto indica que Finesse no está configurado actualmente como suscriptor en este servidor 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
```

Acerca de esta traducción
Cisco ha traducido este documento combinando la traducción automática y los recursos humanos a fin de ofrecer a nuestros usuarios en todo el mundo contenido en su propio idioma.

Tenga en cuenta que incluso la mejor traducción automática podría no ser tan precisa como la proporcionada por un traductor profesional.

Cisco Systems, Inc. no asume ninguna responsabilidad por la precisión de estas traducciones y recomienda remitirse siempre al documento original escrito en inglés (insertar vínculo URL).