Informazioni su Connessione WebSocket per Finesse

Sommario

Introduzione

In questo documento viene descritta completamente la connessione WebSocket in modo che, durante la risoluzione dei problemi, i processi sottostanti siano perfettamente compresi.

Prerequisiti

Requisiti

Nessun requisito specifico previsto per questo documento.

Componenti Utilizzato

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

- Cisco Finesse
- UCCX

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

Web Socket è una connessione permanente tra il client e il server.

Socket Web

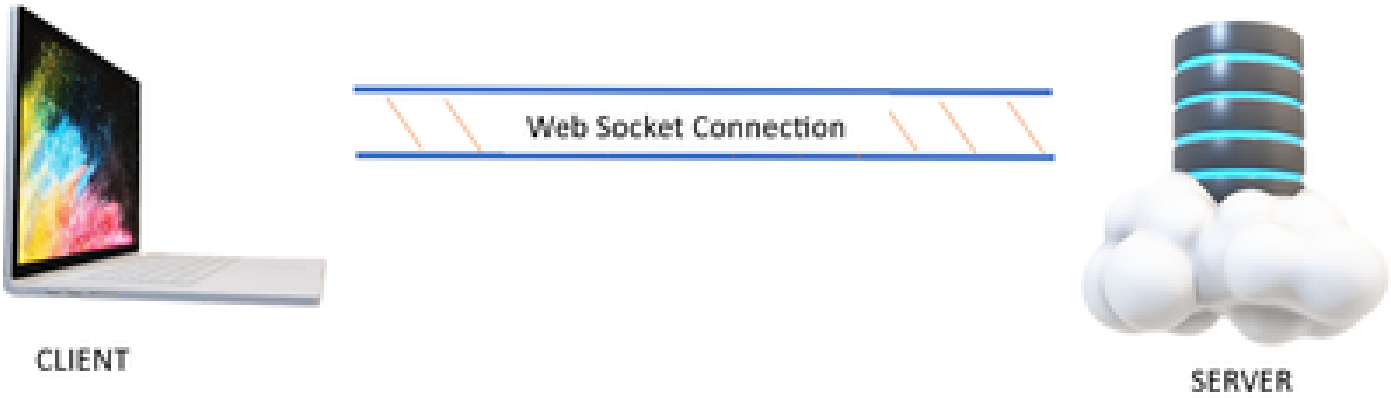
Che cosa si intende per connessione persistente?

Ciò significa che una volta stabilita la connessione tra il client e il server, il client e il server possono inviare e/o ricevere dati in qualsiasi momento.

Questa è una connessione full duplex bidirezionale.

Il server non deve attendere che la richiesta del client esegua il push dei dati.

Analogamente, il client non deve creare ogni volta una nuova connessione per inviare nuovi dati al server.

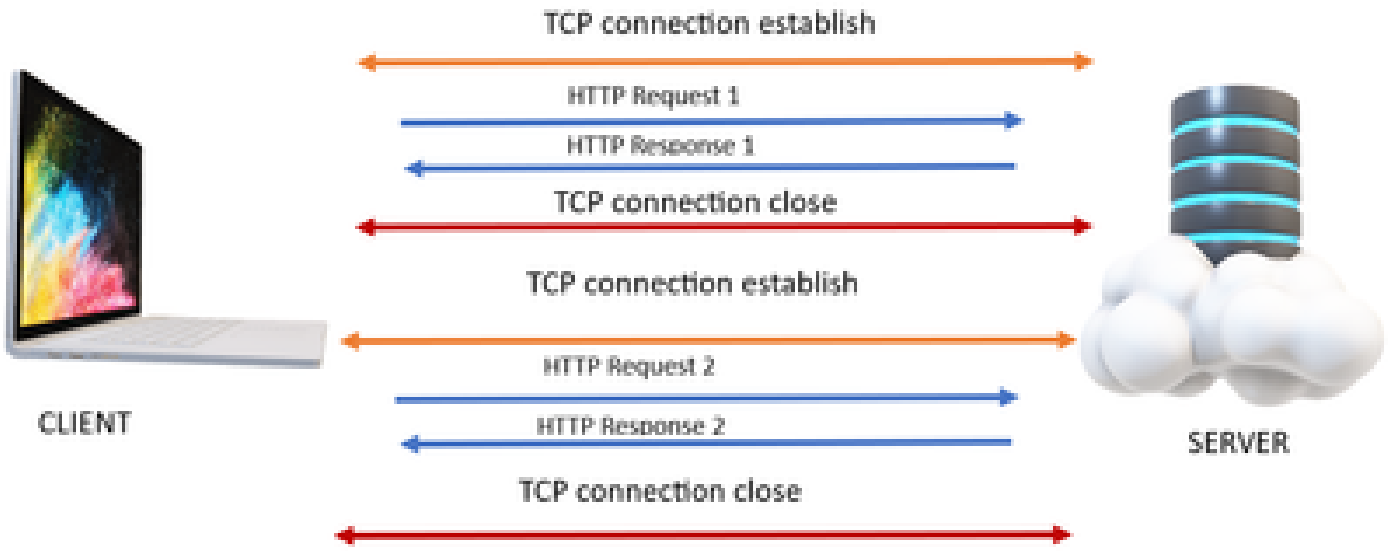


La connessione Web Socket viene utilizzata principalmente nelle applicazioni che richiedono aggiornamenti dei dati in tempo reale.

Ad esempio, le app di trading di titoli, di messaggistica e, nel nostro caso, Cisco Finesse.

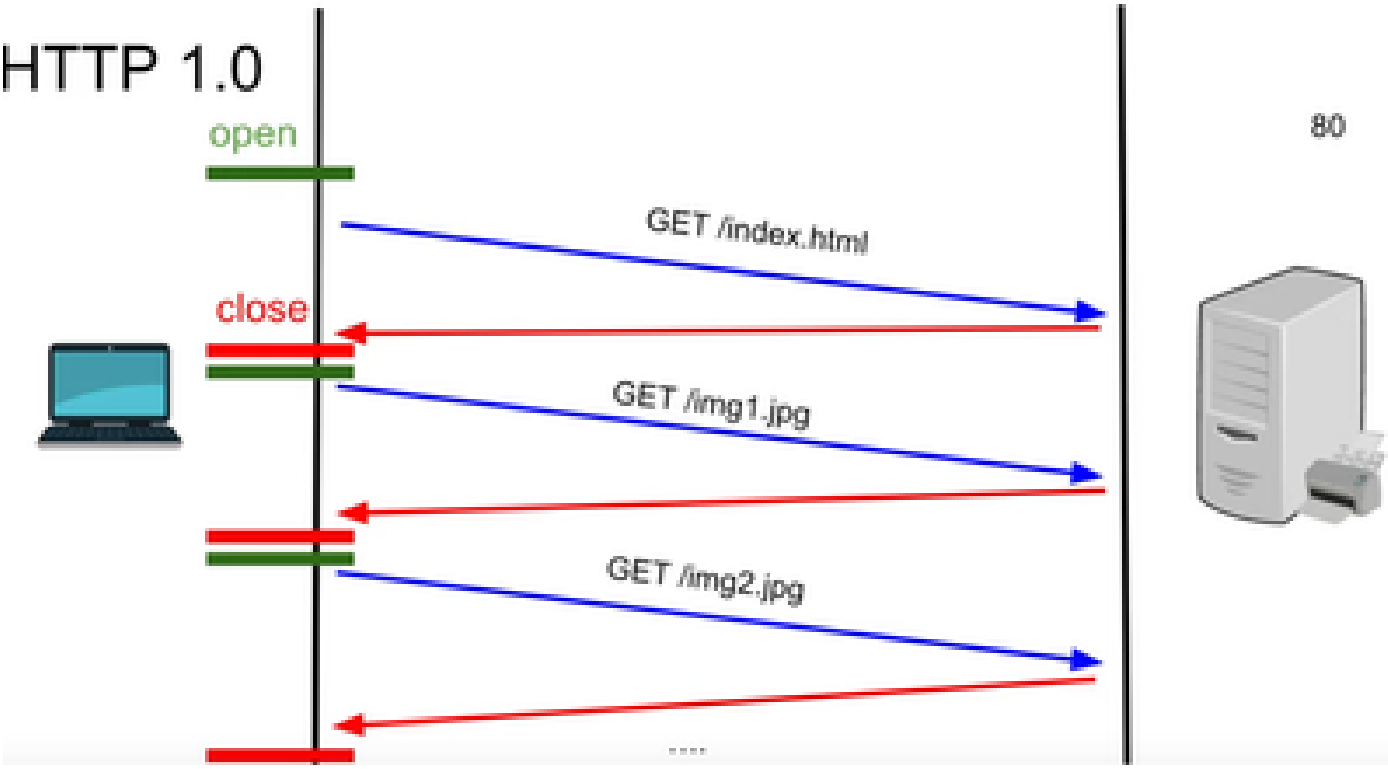
Funzionamento di WebSockets

Considerare:

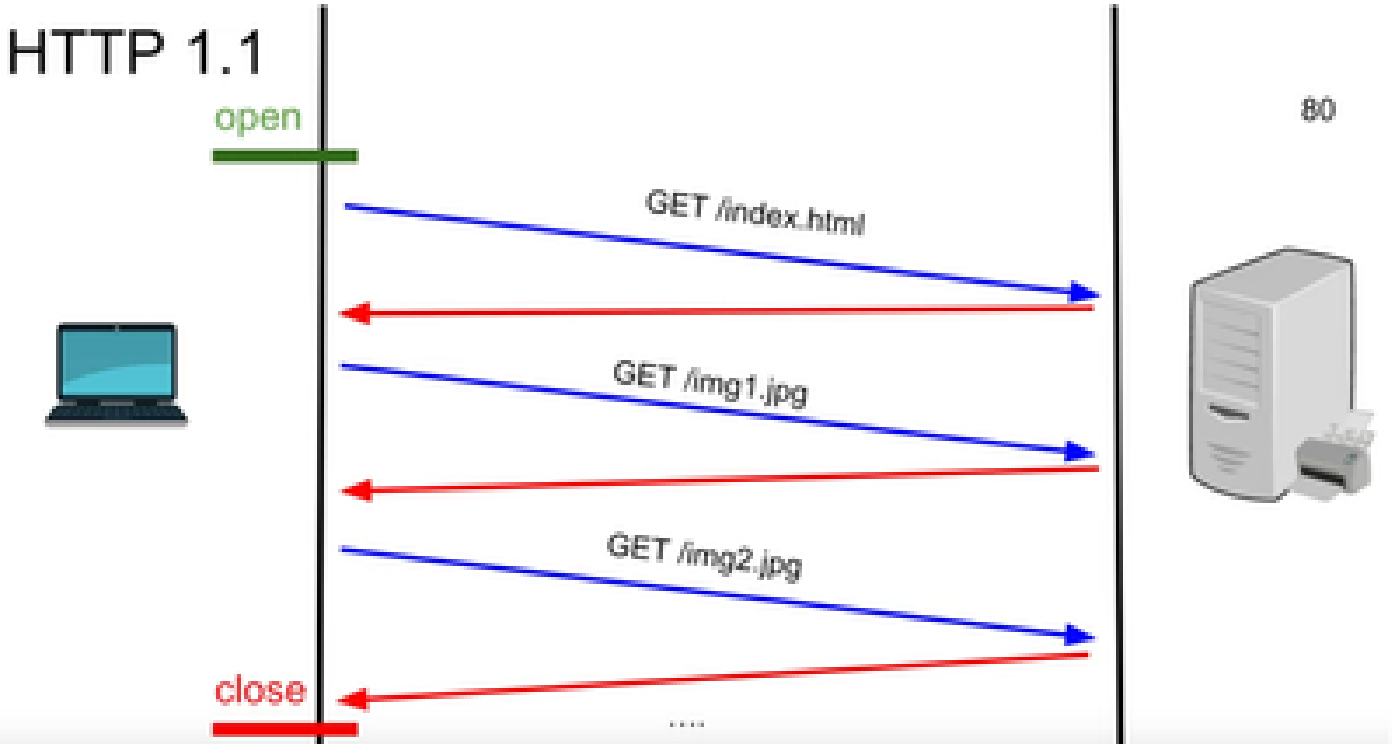


HTTP

1. Viene eseguita la connessione TCP (handshake a 3 vie).
2. Il client invia quindi una richiesta HTTP.
3. Il server invia la risposta HTTP.
4. Dopo un ciclo di risposta alla richiesta, la connessione TCP viene chiusa.
5. Per una nuova richiesta HTTP, la connessione TCP viene stabilita per prima.



HTTP 1.0 - Dopo ogni risposta alla richiesta, l'handshake TCP viene riavviato per un'altra risposta alla richiesta HTTP.



HTTP 1.1 - Questa connessione ha funzionato perché è possibile inviare e ricevere dati e quindi chiudere la connessione.

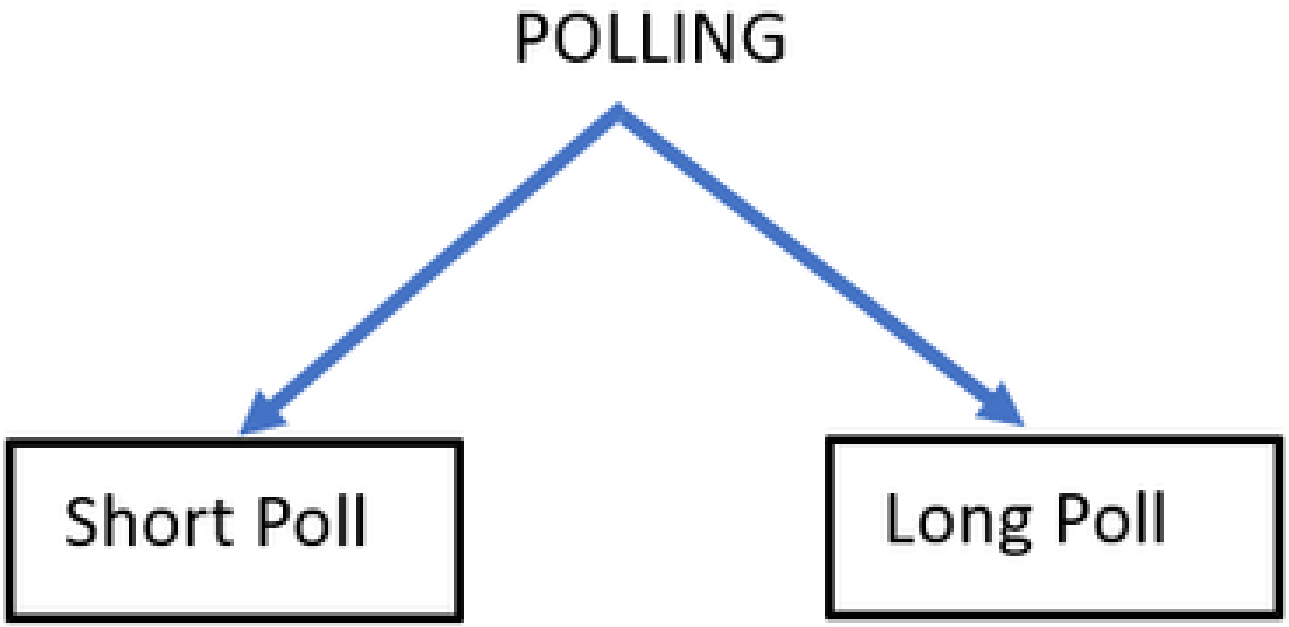
Anche in questo caso, non è adatto per le app in tempo reale perché il server può inviare alcuni dati anche quando il client non li richiede. Pertanto, questo modello non è efficace.

Problema con HTTP

Il problema comincia con i sistemi in tempo reale.

Per un sito Web che richiede aggiornamenti in tempo reale, è molto difficile inviare ogni volta richieste HTTP per ottenere un aggiornamento dal server e che utilizza molta larghezza di banda e causa sovraccarico.

Per risolvere questo problema, viene utilizzato un meccanismo di HTTP denominato Polling.



Sondaggio breve: viene implementato quando viene impostato un timer fisso breve per le richieste e le risposte. Ad esempio, 50 secondi o 1 secondo a seconda dell'implementazione.

Se non è disponibile alcun aggiornamento dall'altra parte, è possibile ottenere risposte vuote in tale intervallo di tempo che possono comportare uno spreco di risorse.

Sondaggio lungo - In qualche modo supera il sondaggio breve, ma ha ancora un tempo fisso per aspettare una risposta.

Se in tale intervallo di tempo non è presente alcuna risposta che è relativamente più lunga del polling breve ma è comunque fissa, richiedere nuovamente i timeout.

Quindi, il voto non è il modo migliore per superare questo problema.

L'altro metodo da utilizzare è SSE.

SSE

Eventi inviati dal server

In questo caso, esiste una connessione unidirezionale tra il server e il client tramite la quale il server può inviare i dati al client in qualsiasi momento.

È importante notare che si tratta di una connessione unidirezionale, il che significa che solo il server può inviare i dati al client e non viceversa.

Un esempio di caso di utilizzo è rappresentato dalle notifiche o dagli aggiornamenti in blocco da un server a un client. Ad esempio, notizie aggiornamenti in tempo reale, Instagram live e così via.

Questa funzionalità non è molto utile per le applicazioni che utilizzano aggiornamenti e messaggi

in tempo reale.

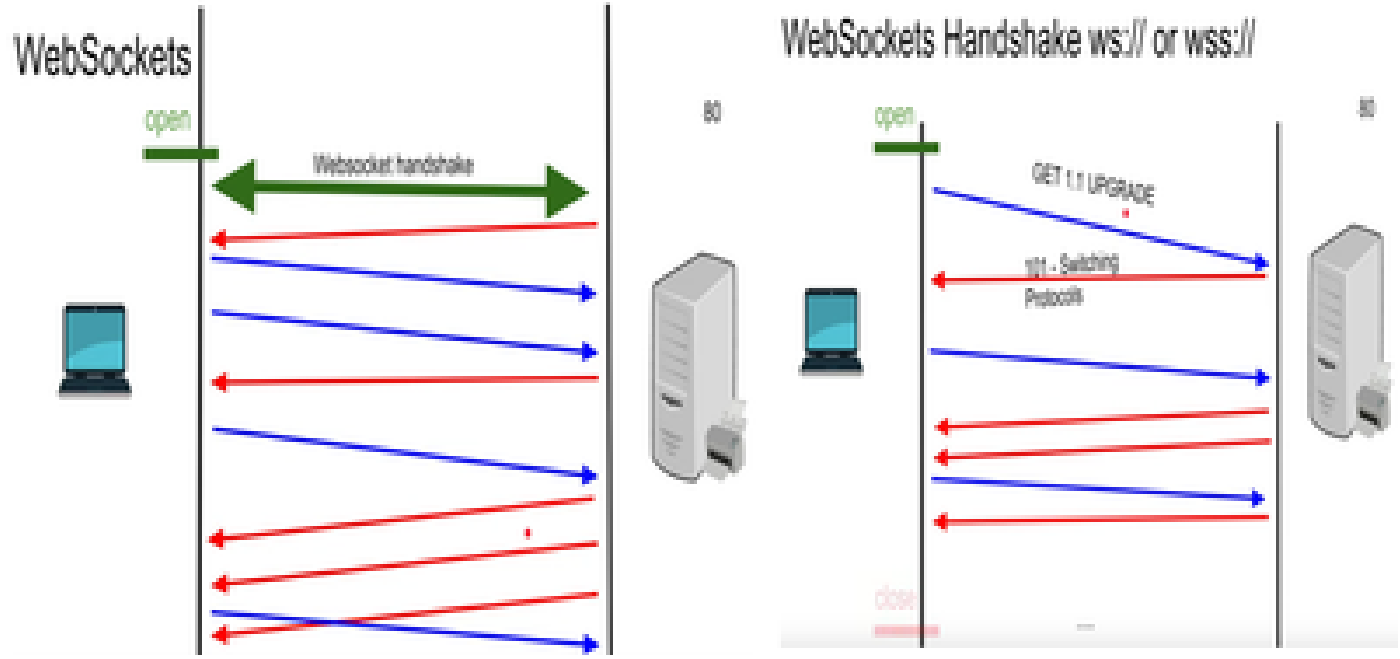
La connessione socket Web è una connessione full duplex bidirezionale permanente.

Potrebbe trattarsi di una telefonata tra un server e un client in cui qualsiasi utente può comunicare con l'altro in qualsiasi momento.

Azioni WebSocket

1. Per stabilire una connessione a un socket Web, il client invia una richiesta di handshake HTTP con un'intestazione aggiornata o aggiornata.
    1. Ciò significa che il client sta dicendo al server che ora, questo è su HTTP ma d'ora in poi, si sposta sulla connessione websocket.
2. Il server risponde quindi con la risposta HTTP 101, che indica che il sito sta cambiando la risposta del protocollo.
3. In seguito, viene stabilita la connessione al socket Web.

Ora il server e il client possono utilizzare la stessa connessione per trasferire i dati l'uno all'altro in qualsiasi momento.



Debug WebSocket

Se a questo punto si accede al client Finesse e si visualizzano i debug di rete, viene visualizzato come:

METODO - GET
Dominio - NOME SERVER

FILE - /WS/

INITIATOR - Openfire.js - socket

Esame della richiesta e della risposta:

Richiesta

OTTIENI

Schema: wss

host: uccxpub.prabhat.com:8445

nomefile : /ws/

Indirizzo: IP del server uccx

Stato: 101

Switching Protocolli

Versione HTTP/1.1

INTESTAZIONE RISPOSTA

Connessione: aggiornamento

Aggiorna: WebSocket

Request – open

Response

PLAIN

http://jabber.org/protocol/caps" hash="sha-1" node="
https://www.igniterealtime.org/projects/openfire/" ver="k3mOuil8afx3OTZxYy6yxLmFsok="/>

Request - auth

YWRtaW5pc3RyYXRvckB1Y2N4cHViLnByYWJoYXQuY29tAGFkbWluaXN0cmF0b3IAMTIzNA==

Response

Request – XMPP Bind Bind request to Bind the resource which in this case is desktop with a jabber id

desktop

Response – XMPP Bind where User ID is given a jabber id

administrator@uccxpub.prabhat.com/desktop

administrator@uccxpub.prabhat.com/desktop

Presence request

Presence response

http://jabber.org/protocol/caps" hash="sha-1" node="
http://www.igniterealtime.org/projects/smack" ver="NfJ3flI83zSdUDzCEICtbypursw=">

http://jabber.org/protocol/caps" hash="sha-1" node="
http://www.igniterealtime.org/projects/smack" ver="NfJ3flI83zSdUDzCEICtbypursw=">

PUBSUB request – Requesting to subscribe the user to the pubsub node so that all the events on the user are monitored.

Response – user subscribed.

http://jabber.org/protocol/pubsub">

PUBSUB request – Requesting to subscribe the Team to the pubsub node so that all the events on the team are monitored.

Response – Team subscribed

http://jabber.org/protocol/pubsub">

Informazioni correlate

- Supporto tecnico Cisco e download

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).