UCS FineReader-architectuur begrijpen

Inhoud

Inleiding

In dit document wordt de architectuur van Finesse op een grondige manier beschreven, zodat de onderliggende processen zinvol zijn bij het oplossen van problemen.

Voorwaarden

Vereisten

Cisco raadt kennis van deze tools en functies aan:

JTAPI - API voor Java-telefonie

API - Toepassingsprogrammeerinterface

UCS - Unified contactcenters Express

CUM - Cisco Unified Communications Manager

CTI - Computer Telephony Integration

Gebruikte componenten

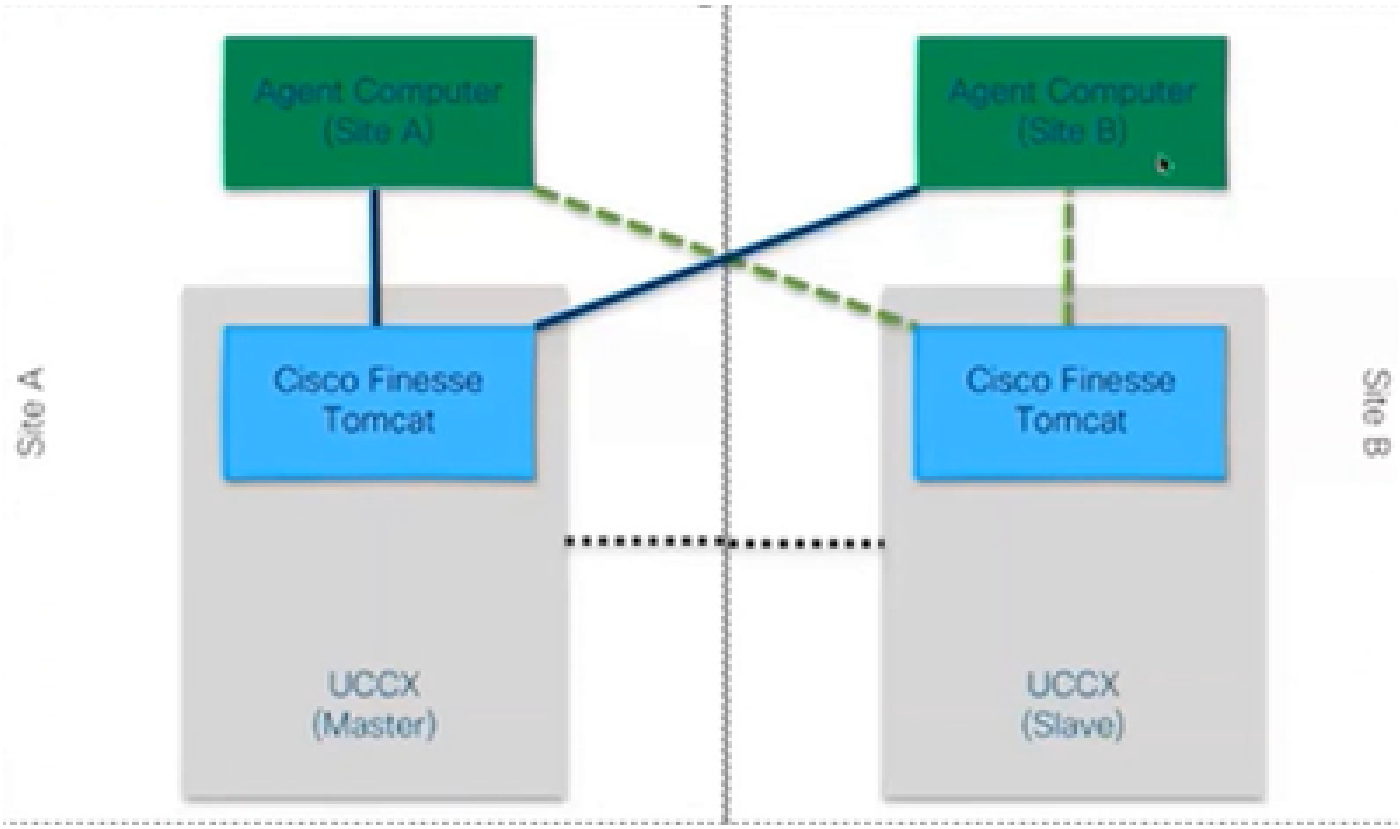
- Cisco Unified Contact Center Express (UCX)

De informatie in dit document is gebaseerd op de apparaten in een specifieke laboratoriumomgeving. Alle apparaten die in dit document worden beschreven, hadden een opgeschoonde (standaard)configuratie. Als uw netwerk live is, moet u zorgen dat u de potentiële impact van elke opdracht begrijpt.

Achtergrondinformatie

In dit document wordt de Finesse-architectuur beschreven, die begint met een overzicht op hoog niveau en vervolgens de signaalstroom in diepte, samen met voorbeelden en diagrammen.

50.000 voet



50000

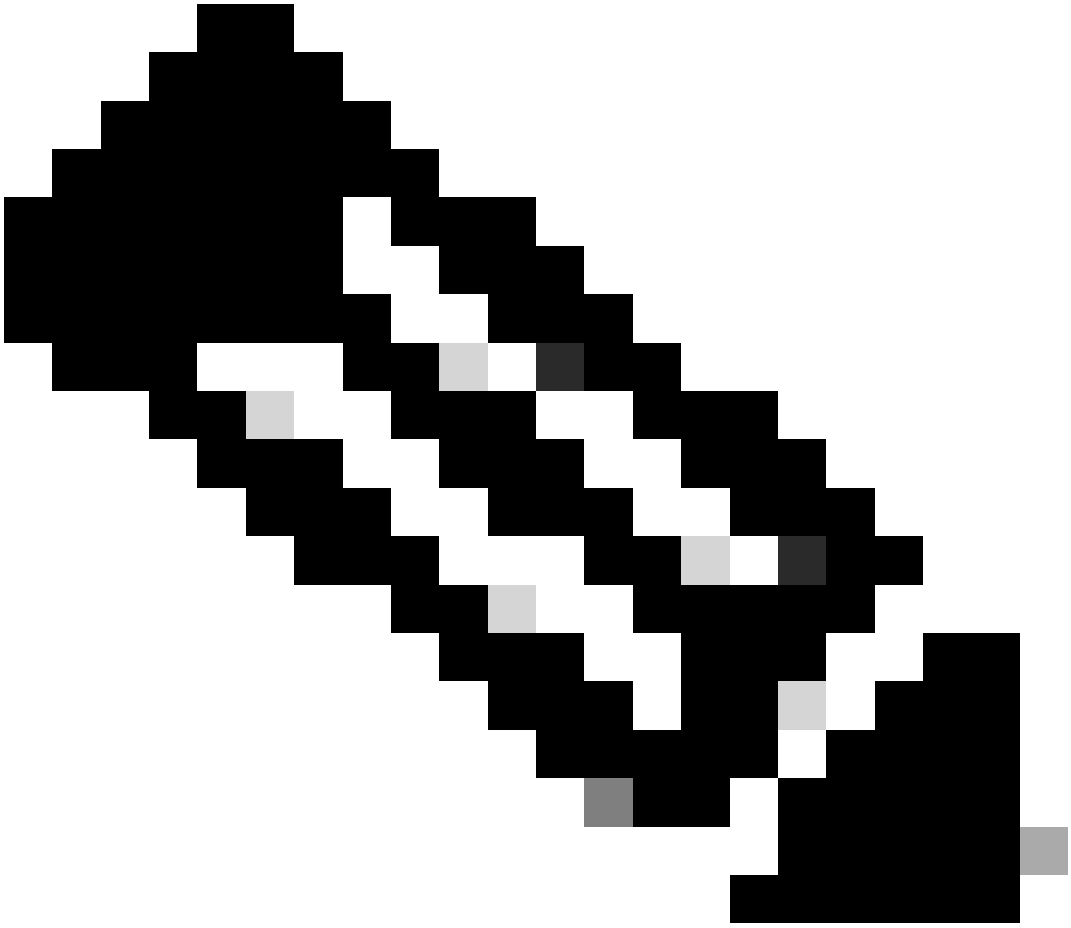
Finesse Tomcat

Finesse Tomcat is vergelijkbaar met Cisco Tomcat in CUCM omdat de functionaliteit hetzelfde is om de webpagina's te laden, maar voor een andere service, genaamd Finesse. Tomcat is ontworpen voor Finesse omdat het een aparte webapplicatie is. U kunt alleen inloggen in finesse met behulp van CCX master node zoals per versies voor 11.5. Vanaf 11.6 kunt u alleen tijdens failover inloggen op een knooppunt (maar niet aanbevolen). Dus als u een WAN-cluster bekijkt, waarbij de agents (op locatie A en op locatie B) zijn verbonden met Finesse op master node, dan is er te allen tijde een inactieve verbinding met het andere knooppunt van Cisco Finesse naar de engine , wat een CTI openconf-verzoek is dat is vereist voor failover.

OpenConf-verzoek wordt regelmatig verzonden om te controleren of het knooppunt actief of stand-by is. Als er een failover is, gebruikt de meldingsservice een API die de SystemInfo API wordt genoemd die de client vertelt dat dit knooppunt is uitgeschakeld en dat er een failover nodig is. Vervolgens wordt er een bestand uitgevoerd dat de browser naar de andere knooppunt leidt en vervolgens wordt de openconf rejavascript verzonden. Deze failover werkt alleen als de certificaten worden geaccepteerd. (om een beveiligde verbinding met de server tot stand te brengen).

Er worden drie certificaten overgelegd:

- Servicecertificaat voor meldingen voor dat knooppunt.
- Servicecertificaat voor meldingen voor externe knooppunten.
- Servicecertificaat voor externe knooppunt voltooien.



Opmerking: de certificaten voor externe knooppunten moeten worden geaccepteerd om failover te kunnen laten werken.

HTTP(EN)

Het is een protocol op de toepassingslaag voor het verzenden van hypermediadocumenten zoals HTML. Het is ontworpen voor communicatie tussen webbrowsers en webservers. Het is een stateless protocol, wat betekent dat de server geen gegevens bewaart tussen de twee verzoeken. Dit is een clientserverprotocol. Voor UCCX wordt de Finesse-client uitgevoerd in de agent-browser (PC). Het doet een verzoek aan de server met behulp van HTTP. Hier zijn enkele veelvoorkomende HTTP-methoden:

GET - om informatie van een server te krijgen.
POST - om informatie naar een server te verzenden.

PUT - om alles op een server te vervangen.
VERWIJDEREN - om informatie van een server te verwijderen.

Finesse gebruikt systeminfo api verzoeken in het http verzoek. Als u bijvoorbeeld de status van een agent wilt wijzigen, stuurt browser een PUT in plaats van een POST, want als POST wordt verzonden, dan raakt de server verward omdat het 2 staten in de hand heeft, welke te selecteren? PUT vervangt de huidige toestand.

XMPP

Xtensible Messaging and Presence Protocol

Het is een reeks protocollen die voor onmiddellijk overseinen en aanwezigheid worden gebruikt. Alle XMPP-entiteiten worden geïdentificeerd met behulp van hun Jabber-id's of JID's. Een van de verlengingen van dit XMPP-protocol dat voor gadgets wordt gebruikt, staat bekend als PUBSUB.

PUBSUB

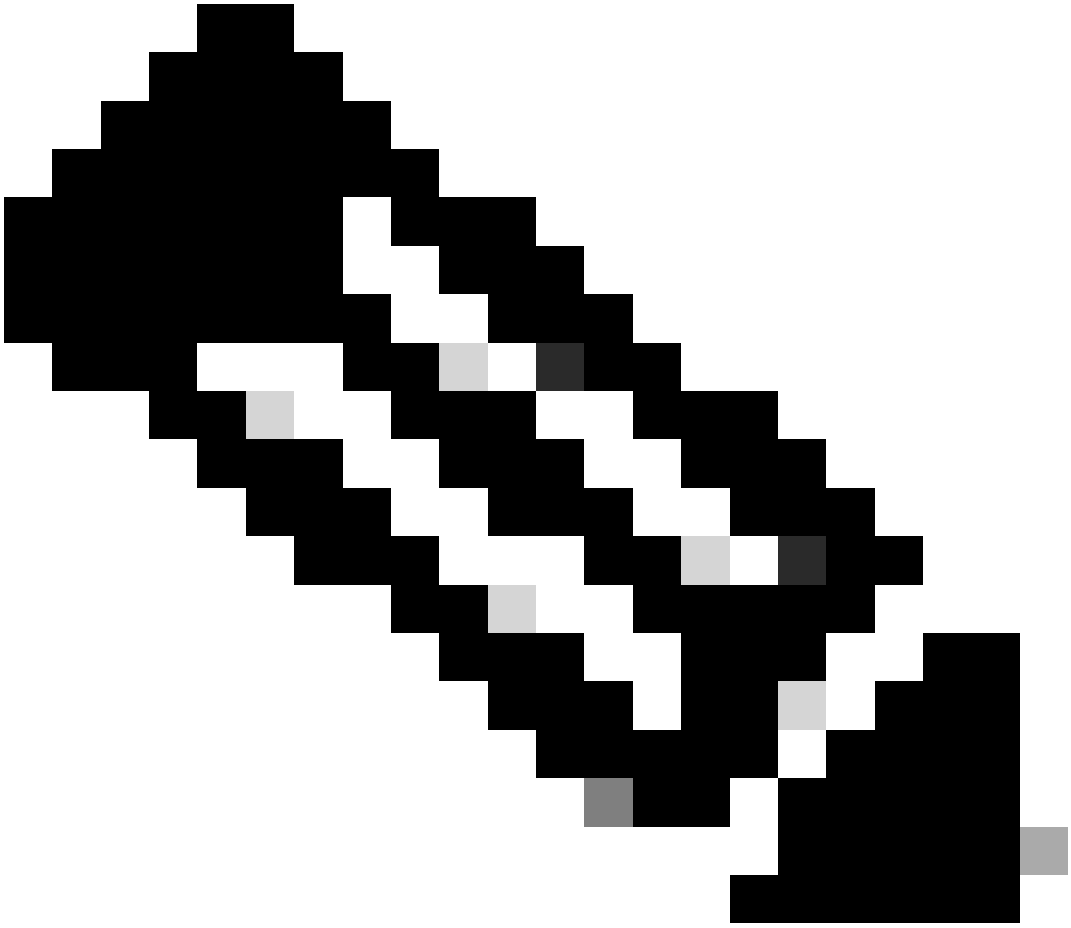
Het is niet de uitgeversabonnee die je kunt bedenken. Het publiceert nog steeds en abonneert nog steeds, maar het heeft niets te maken met de databases. XMPP maakt gebruik van een mechanisme dat knooppunten wordt genoemd. Knooppunt monitort de entiteit waar je om geeft. Alles wat belangrijk voor u is en die u wilt bewaken, voegt u er een knooppunt aan toe. Wat PUBSUB doet, is dat je je abonneert op de updates of meldingen op dat knooppunt. U kunt knooppunten hebben voor elk type entiteit zoals agent, dialoog etc. Als je een knooppunt maakt voor een agent, dan word je geabonneerd op het knooppunt van die agent en wat die agent ook doet, je wordt erover geïnformeerd.

Het doel van deze specificatie is om de XMPP-server (Notification service) in staat te stellen informatie te verkrijgen die is gepubliceerd bij XMPP-knooppunten (onderwerpen) en vervolgens XMPP-gebeurtenissen te verzenden naar entiteiten die zijn geabonneerd op dat knooppunt.

In het geval van Finesse verstuurt de CTI-server (Computer Telephony Integration) CTI-berichten naar de Finesse-webservice om Finesse te informeren over configuratie-updates zoals, maar niet beperkt tot, het aanmaken van een agent of contactserviceroute (CSQ) of informatie over een oproep. Deze informatie wordt vervolgens geconverteerd naar een XMPP-bericht dat de Finesse-webservice publiceert naar de Finesse Notification-service.De Finesse Notification-service stuurt vervolgens XMPP via BOSH-berichten naar agents die zijn geabonneerd op bepaalde XMPP-knooppunten.

BOSH - bidirectionele stromen via synchrone HTTP

BOSH is een langlevende HTTP verbinding waar de server het verzoek voor een langere tijd houdt tot het een reactie op het heeft, anders verstuurt een leeg antwoord. Dit werkt voor XMPP clients en XMPP servers, maar kan ook worden gebruikt voor niet-XMPP toepassingen.



Opmerking: XMPP is stateful terwijl HTTP stateless is (de informatie over het laatste verzoek wordt niet opgeslagen)

Als een webtoepassing met XMPP moet werken, brengt dit meerdere problemen met zich mee.

Probleem 1: Browsers ondersteunen XMPP over Transmission Control Protocol (TCP) niet natief.

Oplossing 1: Webservers en browsers communiceren via HTTP-berichten (HyperText Transfer Protocol), dus Finesse en andere webtoepassingen verpakken XMPP-berichten binnen HTTP-berichten.

Probleem 2: HTTP is een stateless protocol.

Oplossing 2: U kunt hiervoor cookies/postgegevens gebruiken.

Probleem 3: Het derde probleem is het unidirectionele gedrag van HTTP wat betekent dat alleen de client verzoeken verstuurt, en de server kan alleen reageren. De server onmogelijkheid om gegevens te duwen maakt het onnatuurlijk om XMPP via HTTP te implementeren.

Oplossing 3: Om dit probleem te verhelpen, moet u een brug hebben tussen HTTP en XMPP.

De voorgestelde oplossingen zijn:

- Polling (legacy protocol): herhaalde HTTP-verzoeken om nieuwe gegevens gedefinieerd: HTTP-polling

- Lange opiniepeiling is ook bekend als BOSH: transportprotocol dat de semantiek van een langdurende, bidirectionele TCP-verbinding tussen twee entiteiten nabootst door efficiënt gebruik te maken van meerdere synchrone HTTP verzoek/respons paren zonder het gebruik van frequente opiniepeiling te vereisen. De reden om BOSH te gebruiken is om het feit te verdoezelen dat de server niet hoeft te reageren zodra er een verzoek is. De respons wordt vertraagd tot een specifiek moment wanneer de server data heeft voor de client, die dan als respons wordt verzonden. Zodra de client de respons krijgt maakt de client een nieuw verzoek, enzovoorts.

De desktopclient van Finesse (webtoepassing) zet elke 30 seconden een inactieve BOSH-verbinding op via TCP-poort 7443. Als er na 30 seconden geen updates zijn van de Notification Service van Finesse, zal deze een HTTP-respons met 200 OK en een (vrijwel) lege responstekst verzenden. Als de Notification Service een update heeft over bijvoorbeeld de aanwezigheid van een medewerker of dialooggebeurtenis (gesprek), worden de gegevens onmiddellijk naar de Finesse-webclient verzonden.

Kort samengevat:

De Finesse-webclient heeft een inactieve HTTP-verbinding (http-bind) met de Finesse-server via TCP-poort 7443. Dit wordt ook wel een BOSH long poll genoemd. De Finesse Notification Service is een aanwezigheidsdienst die updates post met betrekking tot de staat van een agent, vraag, etc. Als de Notification Service een update heeft, reageert deze op het http-bind-verzoek met de toestandsupdate als een XMPP-bericht in de tekst van de HTTP-respons. Als er 30 seconden na ontvangst van het http-bind-verzoek geen statusupdates zijn, antwoordt de Notification Service zonder toestandsupdates zodat de Finesse-webclient nog een http-bind-verzoek kan verzenden. Dit is een manier voor de Notification-service om te weten dat de Finesse-webclient nog steeds in staat is om verbinding te maken met de Notification Service en dat de agent hun browser niet heeft gesloten of hun computer niet in de slaapstand heeft gezet, enzovoort.

CTI

U kunt Computer Telephony Integration (CTI) gebruiken om voordeel te halen uit computerverwerkingsfuncties bij het maken, ontvangen en beheren van telefoongesprekken. Met CTI-toepassingen kunt u dergelijke taken uitvoeren, zoals het ophalen van gebruikersinformatie uit een database met behulp van een beller-ID, of werken met de informatie die is verzameld door een Interactive Voice Response (IVR)-systeem om een oproep van een gebruiker samen met hun informatie te leiden naar de juiste servicevertegenwoordiger. CTI Manager op CUCM reageert op de JTAPI-verzoeken van UCCX. De CTI server TCP poort is 12018. Zo praten Finesse Server en Engine (CTI Server) met elkaar.

Hier is een deel van de informatie die via CTI wordt uitgewisseld:

- Huidige systeemconfiguratie en toekomstige updates.

- Agenten en hun staten.

- Oproepen en hun staten.

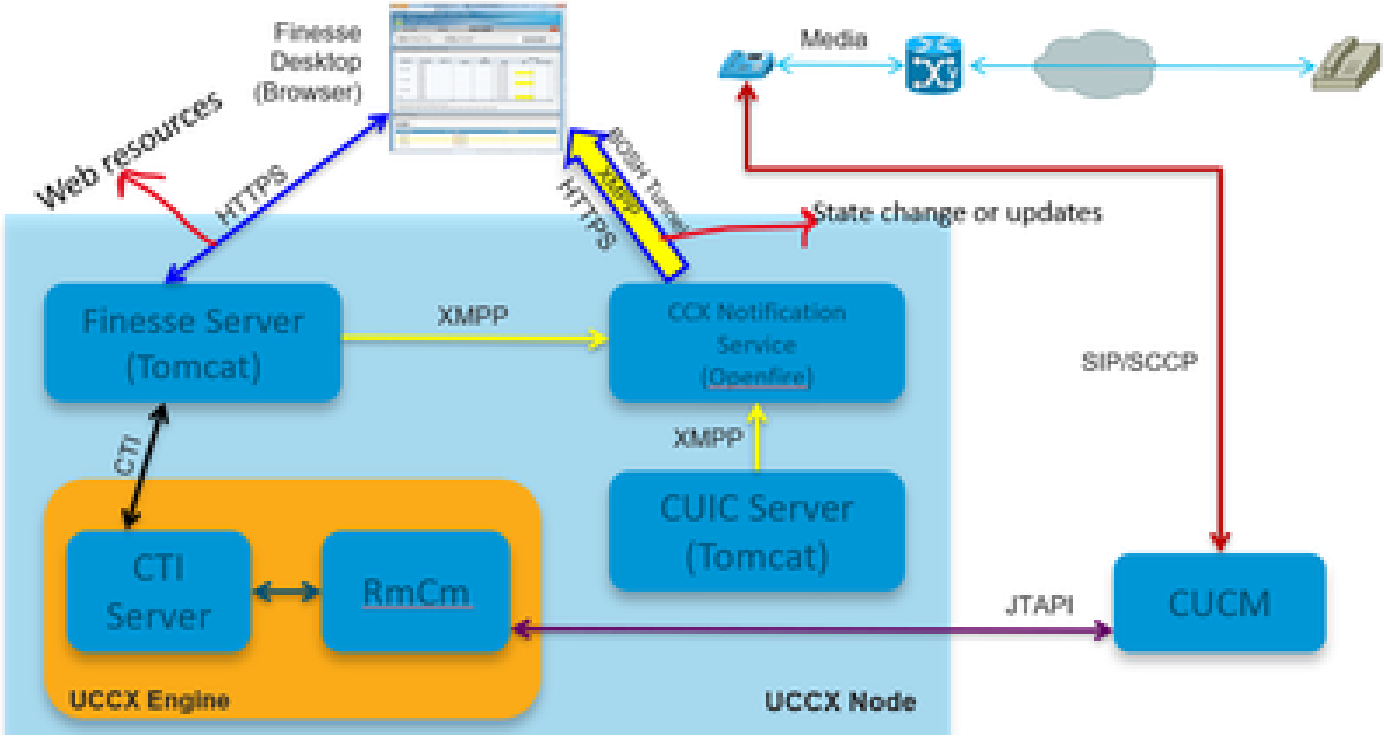
- Stats voor agenten, oproepen en wachtrijen in real time.

JTAPI

Cisco Unified JTAPI fungeert als een programmeerinterfacestandaard die door Sun Microsystems is ontwikkeld voor gebruik met op Java gebaseerde toepassingen voor computertelefonie. Cisco JTAPI implementeert de Sun JTAPI 1.2 specificatie met extra Cisco-extensies. Alle communicatie tussen UCCX en CUCM vindt plaats op JTAPI. Dit is hoe CUCM en Engine (Telephony subsysteem) met elkaar praten. JTAPI wordt gebruikt om CUCM-telefoons, routeoproepen te controleren en te bewaken met behulp van CTI-poorten en routepunten, opnamen te starten en te stoppen op CUCM en voor elke call routing-functionaliteit

30000 voet

In het volgende diagram wordt beschreven hoe de UCCX Engine, Finesse, CUCM en Browser met elkaar communiceren.

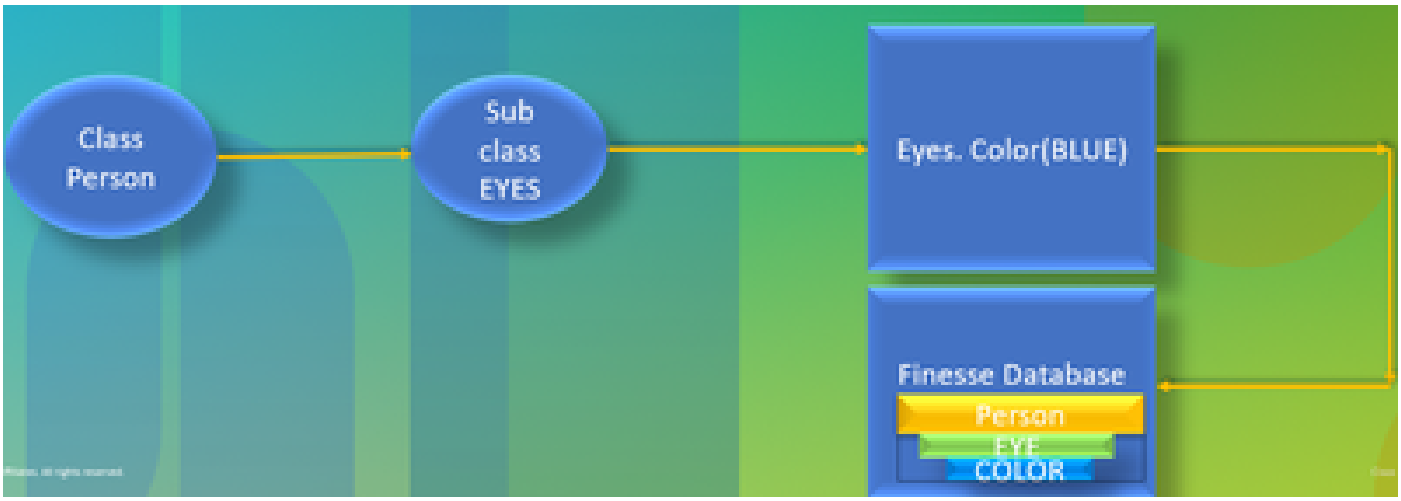


30000

Laten we er rekening mee houden dat de oproep wordt gedaan met de gemachtigde. Nu, RmCm die de agentenuitbreiding door JTAPI controleert, vertelt de server CTI over de staatsverandering dat de agent spreekt. Deze informatie wordt verzonden van de CTI-server (binnen CCX Engine) naar de Finesse-server (Tomcat) met behulp van CTI. Finesse-server stuurt deze informatie naar de CCX-berichtenservice met behulp van XMPP over de statuswijziging. Berichtgevingsservice (Openfire) opent een BOSH-tunnel naar de browser van de agent en werkt de informatie over de statuswijziging bij. Zo ziet u hoe de agent naar GERESERVEERDE status gaat. Om het even welk type van Webmiddelen wordt gevraagd aan de fijnste server die HTTPS zoals de dossiers van de OORLOG, gadgets gebruiken etc. (als niet reeds in geheim voorgeheugen).

HIBERNEREN

In het volgende diagram wordt uitleg gegeven over de Hibernate service.



hiberneren

Hibernate wordt ook wel High-Performance Object/Relational Persistence and Query Service genoemd. Eenvoudig gesteld, brengt het JAVA-klassen aan Databasetabellen in kaart. Bijvoorbeeld, je hebt een JAVA object genaamd Team en je hebt een database tabel in de finesse database genaamd Team. De JAVA-klasse bepaalt welke informatie zich in de tabel bevindt en HIBERNATE is wat dat doet gebeuren. In plaats van SQL queries te gebruiken, maakt het gebruik van java-klassen om de informatie bij te werken.

AXL

Administratieve XML.

XML staat voor eXtensible Markup Language en is een opmaaktaal die relatief eenvoudige regels definieert voor het coderen van gegevens. Het was in de eerste plaats ontworpen om gestructureerde gegevens te verzenden en ontvangen in een welomschreven formaat dat beide systemen kunnen begrijpen. In de meest fundamentele vorm definieert XML tags die zijn ingesloten in haakjes (<>) en deze tags omringen de gegevens die door de tag worden beschreven. Tags kunnen een hiërarchie vormen met tags binnen van andere tags. Bijvoorbeeld, om een basis telefoonapparaat te bepalen, kunt u zeggen dat een telefoonapparaat drie parameters, een naam, een beschrijving, en een telefoonaantal nodig heeft.

Het is een op SOAP gebaseerde API die de externe provisioning op CUCM mogelijk maakt. Het wordt gebruikt om informatie toe te voegen, bij te werken, te verwijderen of terug te halen uit de CUCM-database. De mogelijkheden voor het ophalen zijn onder andere het controleren van gebruikersverificatie en het uitvoeren van SQL queries.De AXL API biedt u toegang tot de gehele CUCM-database. De AXL API is uitsluitend bedoeld voor provisioning en biedt geen toegang tot gegevens over de uitvoering of de prestaties.

AXL API maakt gebruik van de basisverificatie van HTTPS. Elke CUCM-gebruiker (Applicatie of Eindgebruiker) heeft lees-/schrijftoegang via AXL als zij lid zijn van de **Standaard CCM Super Gebruikers** toegangscontrolegroep of een groep met de **Standaard AXL API Access** rol toegewezen. Dit betekent dat alle super-user accounts impliciet al toegang hebben tot de AXL API. Om een account te maken dat speciaal is bedoeld voor AXL API-gebruik, moet u eerst een toegangscontrolegroep maken en de **standaard AXL API Access** rol toewijzen aan deze groep. Vervolgens moet u de toepassingsgebruiker associëren met de nieuwe groep. Om alleen-lezen AXL API-toegang te bieden, kunt u een afzonderlijke toegangscontrolegroep maken en alleen de **standaard AXL Alleen-lezen API**-toegangsrol toekennen.

SOAP

Het Simple Object Access Protocol (SOAP) is een manier om informatie door te geven tussen toepassingen in een XML-formaat. De berichten van de ZEEP worden overgebracht van de verzendende toepassing naar de ontvangende toepassing, typisch over een zitting van HTTP. Het

daadwerkelijke bericht van de ZEEP wordt samengesteld uit het element van de Envelop, dat een element van het Lichaam en een facultatief element van de Kop bevat.

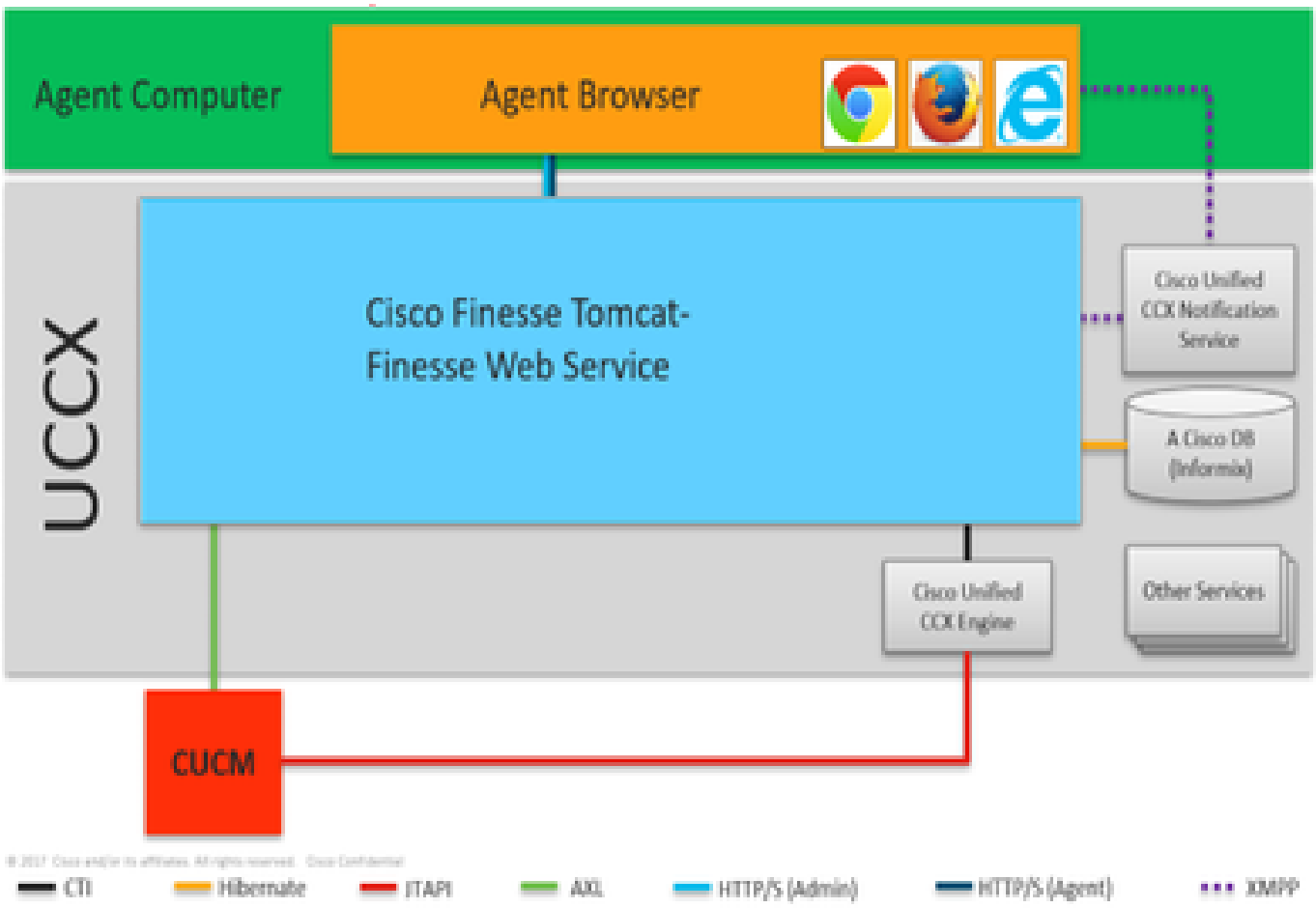
- Envelop - Dit verplichte element is de wortel van het SOAP-bericht, dat de verzonden XML identificeert als een SOAP-pakket. Een enveloppe bevat een body en een optionele header sectie.

- Kop - Dit optionele element biedt een uitbreidingsmechanisme dat verwerkingsinformatie voor het bericht aangeeft. Als de bewerking met het bericht bijvoorbeeld beveiligingsreferenties vereist, moeten deze referenties deel uitmaken van de envelopheader.

- Tekst - Dit element bevat de berichtlading, de ruwe gegevens die worden verzonden tussen de verzendende en ontvangende toepassingen. Het lichaam zelf kan bestaan uit meerdere kindelementen, met een XML-schema dat typisch de structuur van deze gegevens definieert.

20000 voet

In het volgende diagram wordt meer in detail uitgelegd welke protocollen bij de Finse architectuur van belang zijn.



20000

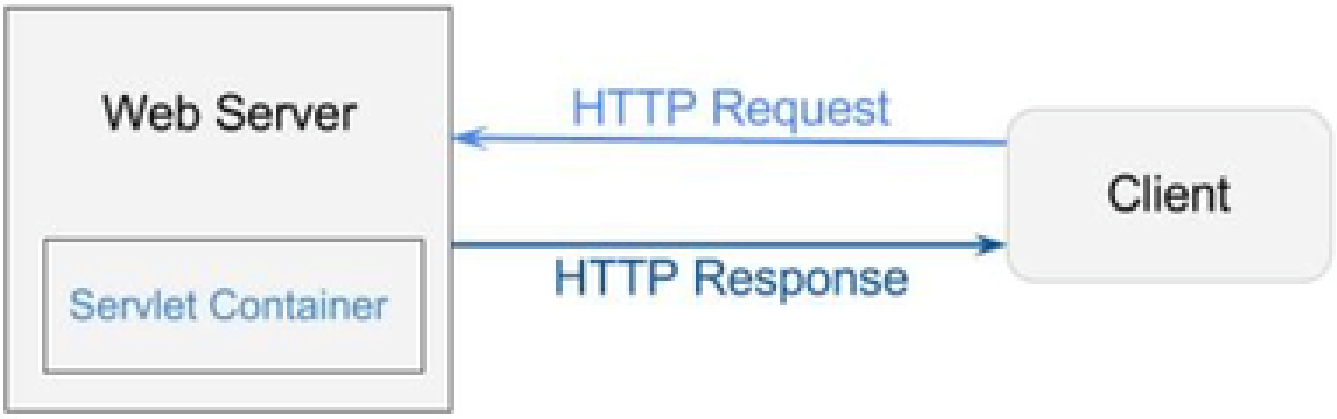
Dit zijn de protocollen die verantwoordelijk zijn voor de communicatie tussen verschillende UCCX-componenten.

- UCS Engine en Finesse Server praten via CTI-protocol.

- UCS Engine en CUCM praten via JTAPI-protocol.

- Finesse Tomcat en CUCM praten over AXL.
- Finesse Notification-service en Agent-browsergesprekken op XMPP/BOSH.
- Finesse Tomcat en de database praten over Hibernate.
- Finesse Tomcat en Finesse Notification praten via XMPP.

APACHE SHINDIG



Shindig

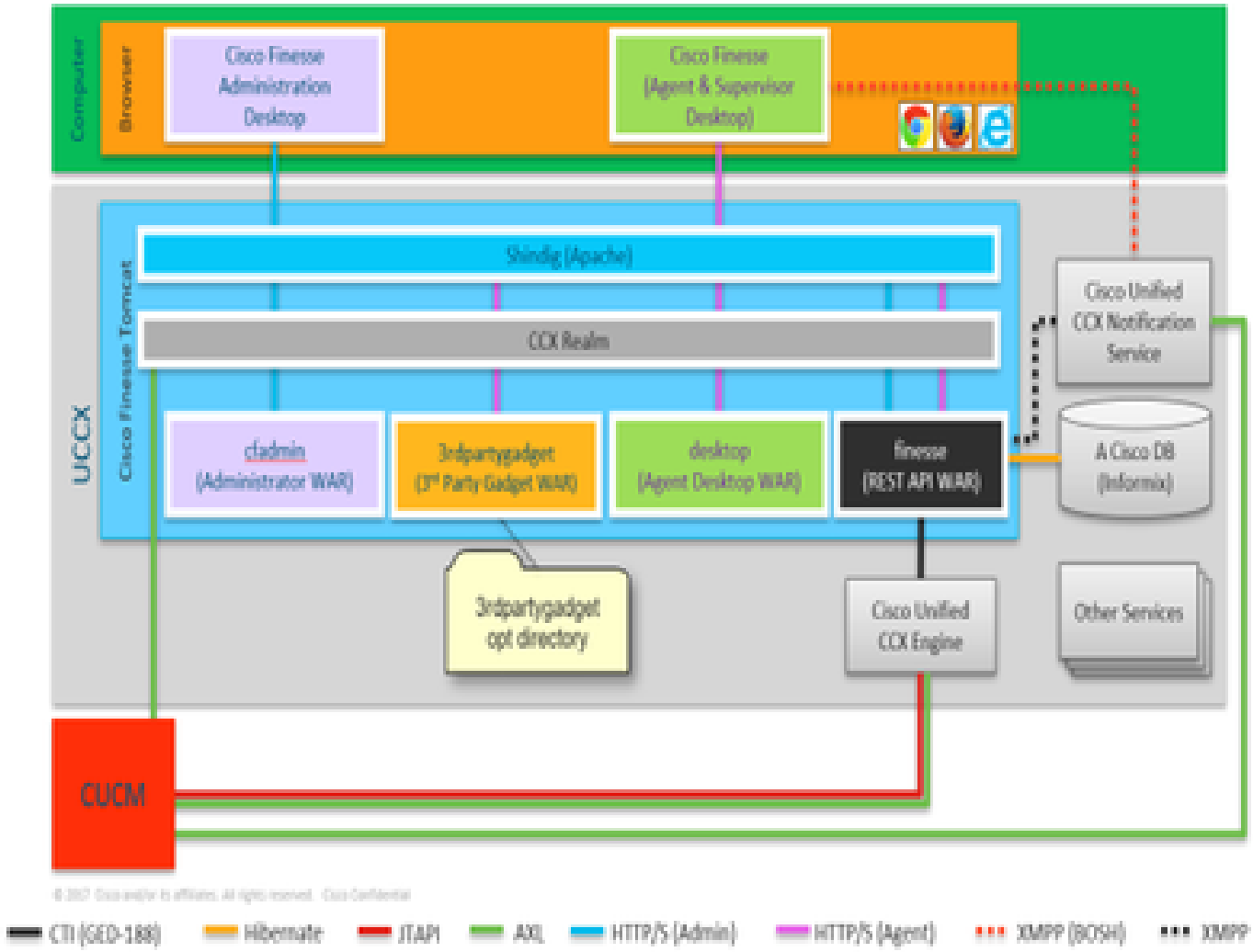
Apache Shindig is een OpenSocial container en helpt u om OpenSocial apps snel te hosten door de code te leveren om gadgets, proxy-verzoeken te renderen en RPC-verzoeken te verwerken. OpenSocial is een verzameling API's voor het bouwen van sociale applicaties die draaien op het web. (Web/Servlet) Container wordt gebruikt door een webserver om dynamisch webpagina's te genereren.

OORLOGSBESTANDEN

WAR staat voor Web Archive. Het bevat bestanden van een webproject. Het kan servlet, XML, JSP, afbeelding, HTML, CSS, JS hebben, enzovoort. Catalina logboeken bevatten de informatie over het inzetten van WARs.

10000 voet

In het volgende diagram wordt in detail uitgelegd hoe de verificatiestroom werkt binnen de componenten van UCCX en Finesse.



10000

WAR-bestanden zijn vereist om de pagina te tonen en aan te maken, afhankelijk van hoe u inlogt. Browser vraagt Shindig dat het een gadget moet renderen, shindig dan praat met CUIC om de gadget te renderen. CCX Realm wordt gebruikt voor verificatie met CUCM met AXL. De meldingsdienst verifieert ook met CUCM die AXL gebruikt.

Finesse Rest API WAR is de belangrijkste repository die communiceert met de notificatieservice, CCX Engine en DB. Shindig praat alleen met Finesse Rest API (WebServices) omdat cfadmin en desktop WARs alleen maar om de pagina weer te geven. Alles wat bij de Finesse Rest API WAR komt, kunt u zien dat in de Finesse WebServices logboeken die de belangrijkste logboeken voor finesse zijn. Je praat over HTTP tussen Shindig en Finesse web service (Rest API WAR). Finesse web service (Rest API WAR) en Engine praten met elkaar via CTI.

AJAX - De schoonheid van Finesse

AJAX staat voor Asynchronous Javascript en XML. Het is geen programmeertaal, maar een methode om toegang te krijgen tot webservers vanaf een webpagina. AJAX is een mechanisme voor het maken van gedeeltelijke pagina updates. Hiermee kunt u delen van een pagina bijwerken met gegevens die van een server komen zonder dat u de hele pagina hoeft te vernieuwen.

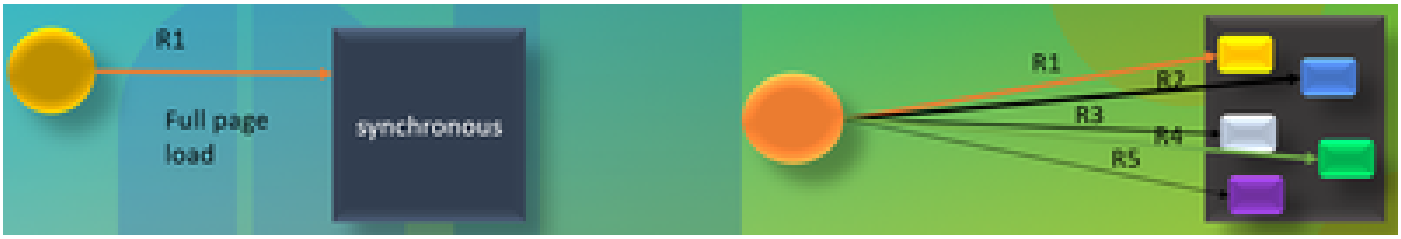
Als je bijvoorbeeld spreekt over Facebook messenger, wanneer een nieuw bericht binnenkomt, hoeft u niet de hele pagina te vernieuwen om het bericht te ontvangen, maar de berichtsectie van de pagina zelf ververst en krijgt de nieuwe berichten in real time zonder dat u de hele pagina hoeft te verversen.

Elke browser heeft een ingebouwd object genaamd XMLHTTPREQUEST (ook wel **XHR** genoemd). Elk verzoek aan AJAX in de server gaat

door dit XML verzoek. Dit bevat de details van wat u moet bijwerken.

Voordelen van AJAX gebruiken

Het volgende diagram verklaart het verschil tussen asynchrone en synchrone verzoeken.



AJAX

In het geval van een synchrone aanvraag, moet u wachten op de eerste aanvraag om verwerkt te worden en dan kunt u een tweede aanvraag verzenden. Pagina's moeten bijvoorbeeld worden ververst en u kunt niets doen totdat de pagina is ververst. Anderzijds, in het geval van een asynchrone aanvraag, hoeft u niet te wachten op het eerste verzoek om voltooid te worden om het tweede verzoek te verzenden. U kunt meerdere aanvragen tegelijkertijd verzenden. Bijvoorbeeld, weer app gadgets op websites. U kunt de weersectie van de pagina en ondertussen ook werken aan de andere secties van de website tegelijkertijd zonder dat u de gehele pagina hoeft te vernieuwen. Dit is het belangrijkste voordeel van Asynchrone aanvraag.

WERKEN VAN AJAX

AJAX is een combinatie van een XMLHTTPREQUEST (**XHR**) die wordt gebruikt om updates te verzenden en ontvangen van webserver samen met Javascript en HTML die worden gebruikt om de gegevens weer te geven of te gebruiken.

AANVRAAG VERZENDEN MET AJAX NAAR SERVER

Dit is een proces in drie stappen dat hierna wordt genoemd:

1. Een variabele maken en het **XHR-**object erin opslaan.

VAR request = nieuwe XMLHttprequest();

2. De aanvraagvariabele met de payload in het XHR-object gebruiken.

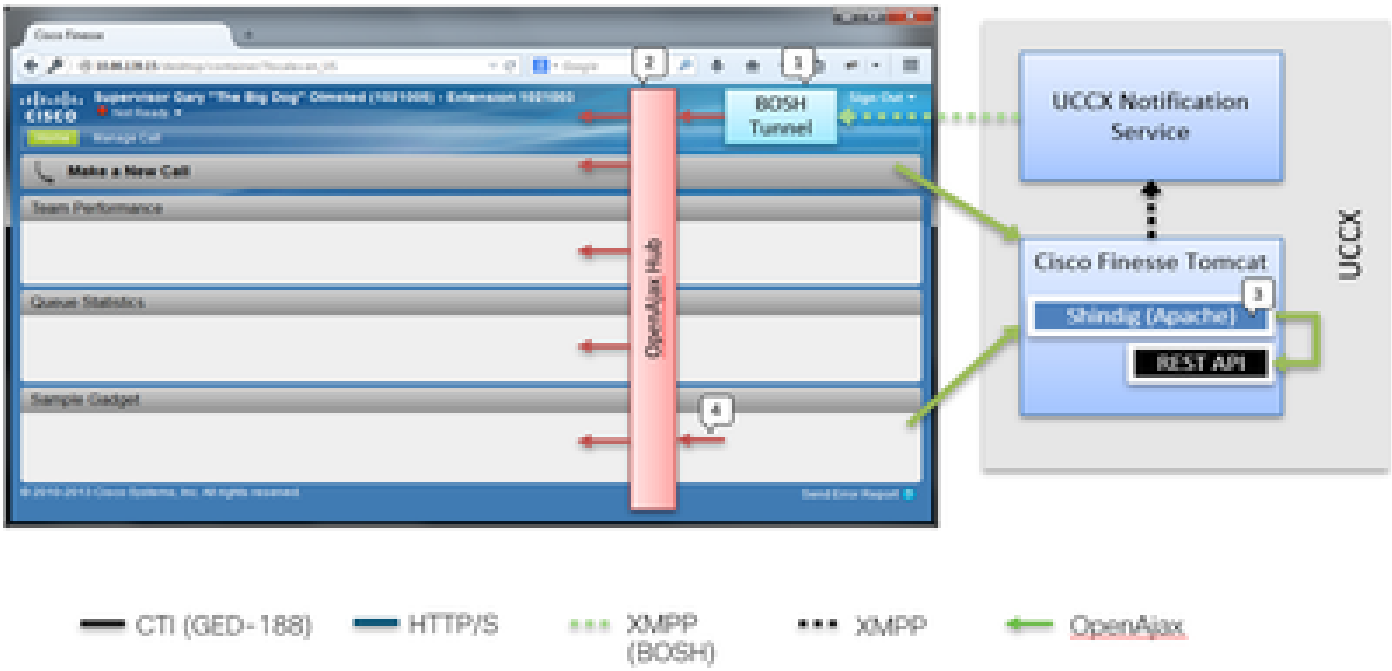
verzoek.**open(GET, URL)**;

3. Verzending van het verzoek

request.send() ;

Desktoparchitectuur

Het volgende diagram verklaart de stroom van AJAX-signalen wanneer gadget op de webpagina wordt weergegeven.

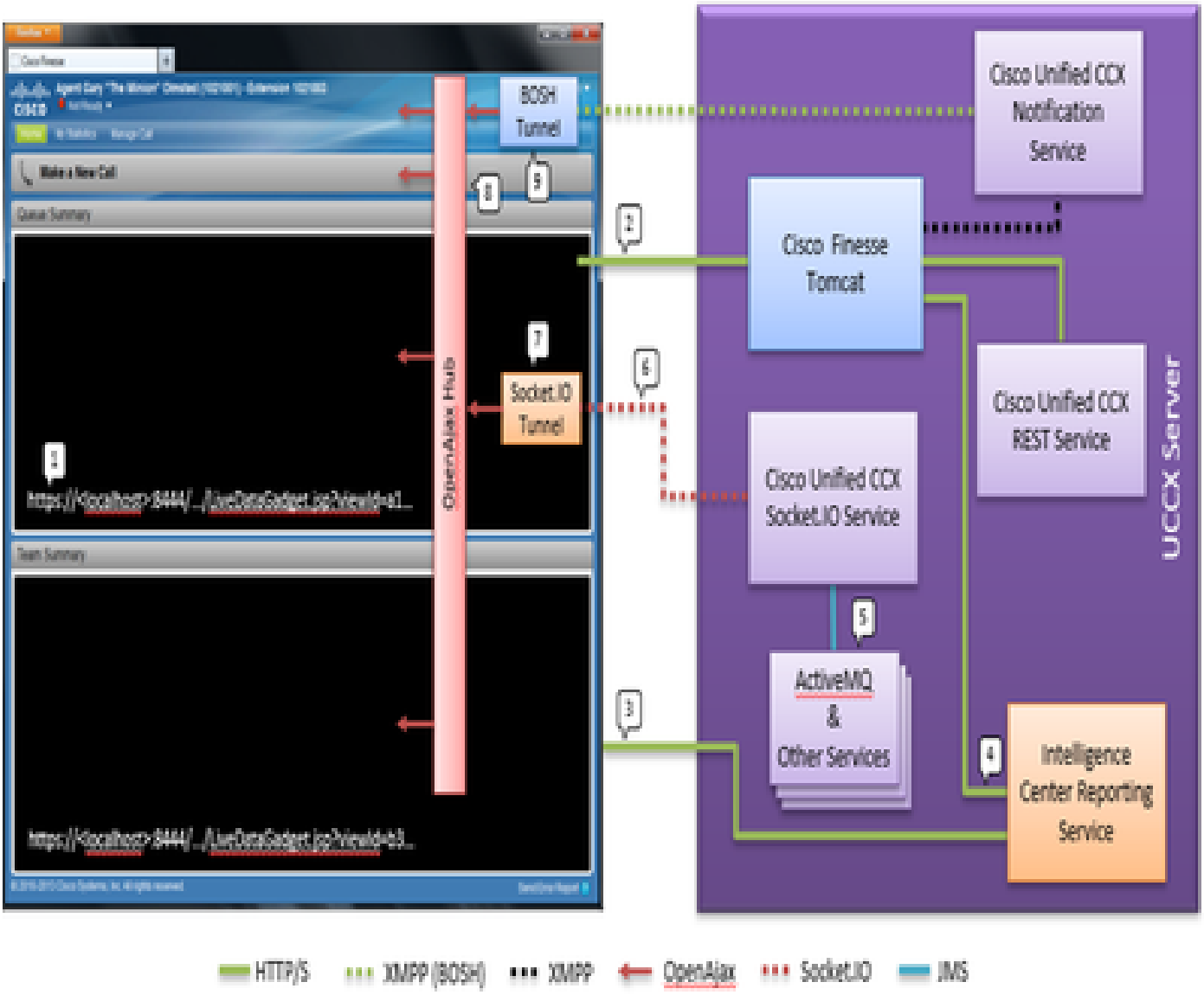


desktoparchitectuur

IFrame bevindt zich in de container om de BOSH Tunnel te hosten. OPENAJAX hub wordt geleverd om berichten te publiceren over de gadgets (met behulp van de openbare methode) REST verzoeken worden ook benaderd via Shindig naar andere servers. Gadgets kunnen hun eigen berichten publiceren op AJAX hub.

Gadget Architecture

In het volgende diagram wordt de architectuur van Finesse Gadget in detail uitgelegd.



gadget architectuur

In tegenstelling tot typische Gadgets, ontvangt CUIC Gadgets ook een real-time XMPP feed van OpenFire. In het geval van UCCX, waar CUIC en Finesse mede-inwoner zijn van UCCX, is er een gedeelde OpenFire-instantie. Het grootste deel van de Gadget-content en alle REST API's worden benaderd via Shindig in de Finesse Server. Dit geldt voor Finesse Gadgets en REST API evenals CUIC Gadget instanties en REST API. CUIC-gadgets gebruiken een D-Grid voor het weergeven van hun rapporten. Er is een bootstrapping proces dat moet plaatsvinden en dit wordt gedaan in samenwerking met CUIC direct. Daarom praten de CUIC Gadgets in eerste instantie direct met CUIC Server tijdens het laadproces. Om deze reden moet het CUIC-certificaat worden geaccepteerd in de gebruikersbrowser (naast de Socket.IO Tunnel). De inhoud van de gadget en de REST API's zijn proxied aan de cliënt tussen Finesse en CUIC. Rest API-oproepen worden gedaan naar zowel de Intelligence Center Reporting Service als de CCX Web Service. De CCX Live Data Socket.IO Service krijgt de berichten van Live Data via JMS van ActiveMQ. De CCX Live Data Socket.IO Service publiceert Real-Time Reporting JSON via de Socket.IO-verbinding van de client. Net zoals de manier waarop de Finesse Desktop een BOSH Tunnel iFrame heeft dat de BOSH-verbinding met de Cisco Finesse Notification Service onderhoudt, heeft de master Live Data Gadget een Socket.IO Tunnel iFrame dat de Socket.IO (websocket) verbinding onderhoudt met de CCX Live Data Socket.IO Service.

De OpenAjax Hub verdeelt alle gebeurtenissen aan de geabonneerde luisteraars. Dit zou zowel Gadgets als delen van de Finesse Container zelf zijn. Het Finse bureaublad heeft een BOSH Tunnel iFrame dat de BOSH-verbinding met de Cisco Unified CCX Notification Service onderhoudt. Dit publiceert evenementen op de OpenAjax Hub.

Referentie links

- Finesse Web Services Ontwikkelaarshandleiding

Over deze vertaling

Cisco heeft dit document vertaald via een combinatie van machine- en menselijke technologie om onze gebruikers wereldwijd ondersteuningscontent te bieden in hun eigen taal. Houd er rekening mee dat zelfs de beste machinevertaling niet net zo nauwkeurig is als die van een professionele vertaler. Cisco Systems, Inc. is niet aansprakelijk voor de nauwkeurigheid van deze vertalingen en raadt aan altijd het oorspronkelijke Engelstalige document (link) te raadplegen.