

Verständnis und Fehlerbehebung bei Finesse BOSH-Implementierungen

Inhalt

[Einleitung](#)

[Voraussetzungen](#)

[Anforderungen](#)

[Verwendete Komponenten](#)

[Hintergrundinformationen](#)

[Verständnis der Finesse BOSH-Implementierung](#)

[XMPP verstehen](#)

[Beispiel-XMPP-Nachricht](#)

[XMPP-Implementierung mit Finesse](#)

[Finesse XMPP-Anfrage/-Antwort \(Beispiel\)](#)

[Finesse XMPP-Nachrichten und XMPP-Knoten verstehen](#)

[Beispiel 1: Verwenden von Pidgin zum Anzeigen von Finesse XMPP-Knoten](#)

[Beispiel 2: Verwenden der Registerkarte Netzwerk der Browser-Entwicklertools zum Anzeigen von HTTP-Nachrichten](#)

[Fehlerbehebung: Fehlermeldung bei BOSH-Verbindungstrennung](#)

[Protokollanalyse](#)

[Debug Notification Service-Protokolle](#)

[Info Notification Service-Protokolle](#)

[Webservices-Protokolle](#)

[Allgemeine Gründe für BOSH-Trennung](#)

[Problem - Verbindung der Agenten zu verschiedenen Zeiten getrennt \(Client-seitiges Problem\)](#)

[Empfohlene Maßnahmen](#)

[Problem - Alle Agenten trennen gleichzeitig die Verbindung \(serverseitiges Problem\)](#)

[Empfohlene Maßnahmen](#)

[Fiddler verwenden](#)

[Häufiges Fiddler-Problem](#)

[Beispiel für Konfigurationsschritte](#)

[Wireshark verwenden](#)

[Verwandte Fehler](#)

[Zugehörige Informationen](#)

Einleitung

In diesem Dokument wird die Architektur von Finesse-Verbindungen beschrieben, die BOSH verwenden, und es wird beschrieben, wie BOSH-Verbindungsprobleme diagnostiziert werden können.

Voraussetzungen

Anforderungen

Cisco empfiehlt, dass Sie über Kenntnisse in folgenden Bereichen verfügen:

- Cisco Finesse
- Unified Contact Center Enterprise (UCCE)
- Unified Contact Center Express (UCCX)
- Webbrowser-Entwicklungstools
- Windows- und/oder Mac-Administration

Verwendete Komponenten

Die Informationen in diesem Dokument basierend auf folgenden Software- und Hardware-Versionen:

- Cisco Finesse 9.0(1) - 11.6(1)
- UCCX 10.0(1) - 11.6(2)

Die Informationen in diesem Dokument beziehen sich auf Geräte in einer speziell eingerichteten Testumgebung. Alle Geräte, die in diesem Dokument benutzt wurden, begannen mit einer gelöschten (Nichterfüllungs) Konfiguration. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die möglichen Auswirkungen aller Befehle kennen.

Hintergrundinformationen

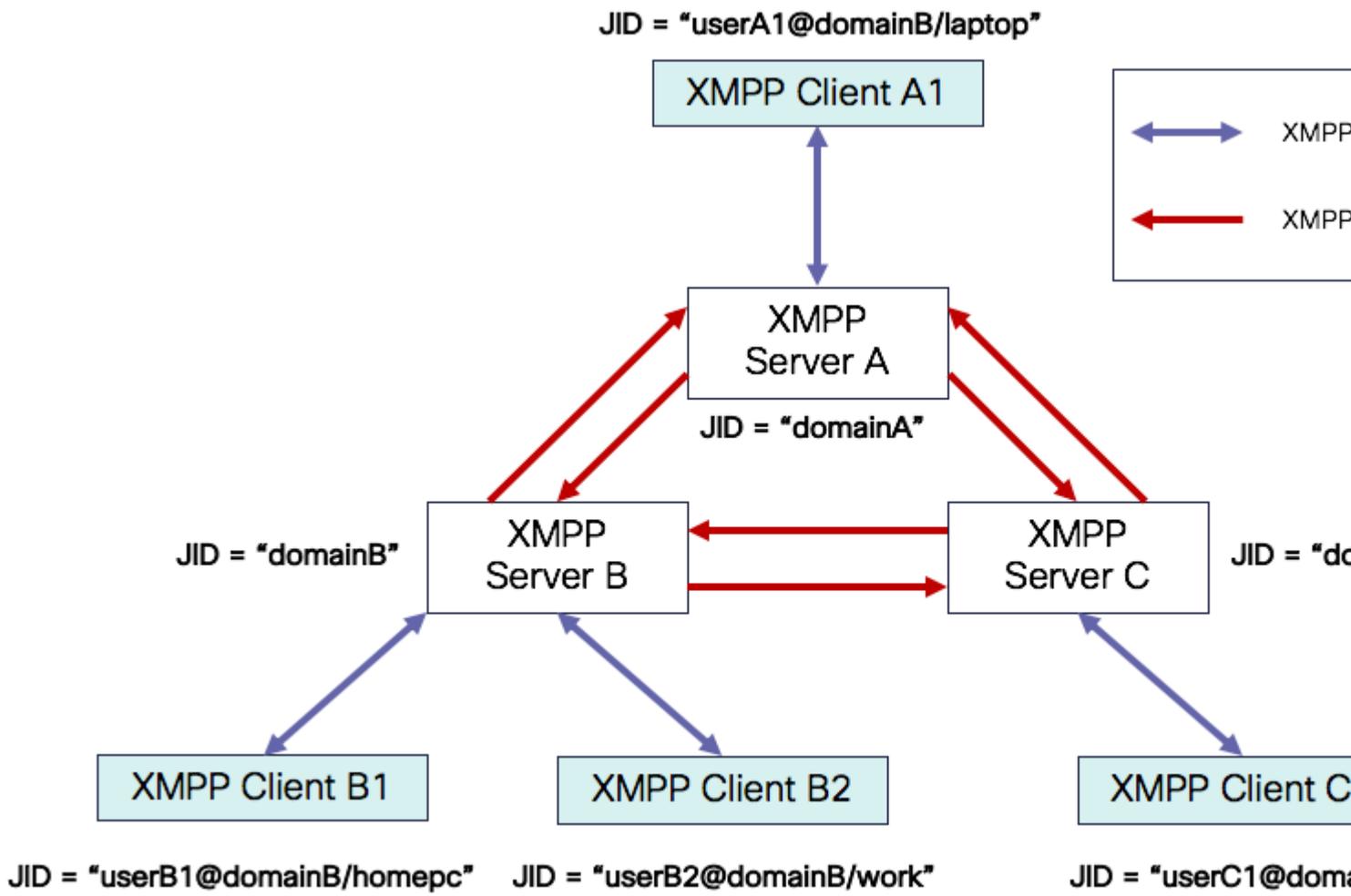
Verbindungen, die bidirektionale Streams über synchrones HTTP verwenden, werden als BOSH bezeichnet.

Verständnis der Finesse BOSH-Implementierung

XMPP verstehen

Extensible Messaging and Presence Protocol (XMPP) (auch als Jabber bekannt) ist ein Stateful-Protokoll in einem Client-Server-Modell. XMPP ermöglicht die schnelle Bereitstellung kleiner strukturierter XML-Daten (eXtensible Markup Language) von einer Einheit an eine andere. XMPP/Jabber wird in großem Umfang in Instant Messaging- (IM) und Presence-Anwendungen eingesetzt.

Alle XMPP-Einheiten werden durch ihre Jabber-ID (JID) identifiziert.



JID-Adressierungsschema: user@domain/resource

Benutzer	Client-Benutzername auf dem XMPP-Server oder Name des Konferenzraums
Domäne	Vollqualifizierter Domänenname (FQDN) des XMPP-Servers
Ressource	Kennung der spezifischen Einheit/des Endpunkts des Benutzers (z. B. Laptop, Smartphone usw.), Sitzungskennung oder Name des öffentlichen Unterknotens

Hinweis: Nicht alle drei JID-Komponenten werden in allen Fällen verwendet. Ein Server wird normalerweise nur durch die Domäne, einen Konferenzraum durch user@domain und einen Client durch user@domain/resource definiert.

XMPP-Nachrichten werden als Strophen bezeichnet. In XMPP gibt es drei Kernstrophen:

1. <Nachricht>: eine Richtung, ein Empfänger
2. <Präsenz>: eine Richtung, zu vielen veröffentlichen
3. <iq>: Info/Abfrage - Anfrage/Antwort

Alle Strophen haben zu und von Adressen und die meisten Strophen haben auch type, id und xml:langattribute.

Stanza-Attribut	Zweck
zu	Ziel-JID
von	Quell-JID
typ	Zweck der Nachricht
ID	eindeutige Kennung zur Verknüpfung einer Anforderung mit einer Antwort für <iq> Strophen
xml:lang	definiert die Standardsprache für jede für Menschen lesbare XML-Datei in der Strophe

Beispiel-XMPP-Nachricht

```
<message to='person1@example' from='person2@example' type='chat'>
  <subject> Team meeting </subject>
  <body>Hey, when is our meeting today? </body>
  <thread>A4567423</thread>
</message>
```

XMPP-Implementierung mit Finesse

Wenn eine Webanwendung mit XMPP arbeiten muss, treten mehrere Probleme auf. Browser unterstützen XMPP über Transmission Control Protocol (TCP) nicht nativ, sodass der gesamte XMPP-Datenverkehr von einem Programm verarbeitet werden muss, das im Browser ausgeführt wird. Webserver und Browser kommunizieren über HTTP-Nachrichten (HyperText Transfer Protocol), sodass Finesse und andere Webanwendungen XMPP-Nachrichten in HTTP-Nachrichten einbinden.

Die erste Schwierigkeit bei diesem Ansatz besteht darin, dass HTTP ein Stateless-Protokoll ist. Das bedeutet, dass jede HTTP-Anfrage nicht mit einer anderen Anfrage in Zusammenhang steht. Dieses Problem kann jedoch mit anwendbaren Mitteln gelöst werden - beispielsweise durch den Einsatz von Cookies/Postdaten.

Die zweite Schwierigkeit ist das unidirektionale Verhalten von HTTP. Nur der Client sendet Anfragen, und der Server kann nur antworten. Da der Server keine Daten per Push bereitstellen kann, ist die Implementierung von XMPP über HTTP unnatürlich.

Dieses Problem tritt in der ursprünglichen XMPP-Core-Spezifikation (RFC 6120), in der XMPP an TCP

gebunden ist, nicht auf. Wenn Sie jedoch das Problem mit XMPP angehen möchten, das beispielsweise an HTTP gebunden ist, da Javascript HTTP-Anfragen senden kann, gibt es zwei mögliche Lösungen. Für beide wird eine Brücke zwischen HTTP und XMPP benötigt.

Die vorgeschlagenen Lösungen sind:

1. Polling (Legacy-Protokoll): wiederholte HTTP-Anfragen, die nach neuen, in XEP-0025 definierten Daten fragen: Jabber HTTP Polling

2. Langes Polling wird auch als BOSH bezeichnet: Transportprotokoll, das die Semantik einer langlebigen, bidirektionalen TCP-Verbindung zwischen zwei Einheiten emuliert, indem mehrere synchrone HTTP-Anfrage/Antwort-Paare effizient verwendet werden, ohne dass häufiges Polling gemäß XEP-0124: HTTP Binding und erweitert durch XEP-0206: XMPP Over BOSH erforderlich ist.

Finesse implementiert BOSH, da es in Bezug auf die Serverauslastung und den Datenverkehr sehr effizient ist. Der Grund für die Verwendung von BOSH besteht darin, die Tatsache zu vertuschen, dass der Server nicht sofort antworten muss, wenn eine Anfrage eingeht. Die Antwort wird bis zu einem bestimmten Zeitpunkt verzögert, bis der Server über Daten für den Client verfügt. Anschließend wird sie als Antwort gesendet. Sobald der Client die Antwort erhält, stellt er eine neue Anforderung usw.

Der Finesse Desktop-Client (Web-Anwendung) stellt alle 30 Sekunden eine veraltete BOSH-Verbindung über den TCP-Port 7443 her. Wenn nach 30 Sekunden keine Updates vom Finesse-Benachrichtigungsdienst vorliegen, sendet der Benachrichtigungsdienst eine HTTP-Antwort mit 200 OK und einem (fast) leeren Antworttext. Wenn der Notification Service z.B. über ein Update bei Vorliegen eines Agenten oder eines Dialogereignisses (Call) verfügt, werden die Daten sofort an den Finesse Web Client gesendet.

Finesse XMPP-Anfrage/-Antwort (Beispiel)

Dieses Beispiel zeigt die erste XMPP-Nachrichtenanforderungsantwort, die vom Finesse-Client und dem Finesse-Server gemeinsam verwendet wird, um die BOSH-Verbindung einzurichten.

Finesse client request:

```
<body xmlns="http://jabber.org/protocol/httpbind" xml:lang="en-US" xmlns:xmpp="urn:xmpp:bosh" hold="1"
```

Finesse server response:

```
<body xmlns="http://jabber.org/protocol/httpbind" xmlns:stream="http://etherx.jabber.org/streams" authi
```

Zusammenfassung:

1. Der Finesse-Web-Client hat eine veraltete HTTP-Verbindung (http-bind) zum Finesse-Server über den TCP-Port 7443 eingerichtet. Dies wird als lange BOSH-Umfrage bezeichnet.
2. Der Finesse Notification Service ist ein Presence-Service, der Updates zum Status eines Agenten, einen Anruf usw. bereitstellt.
3. Wenn der Benachrichtigungsdienst über eine Aktualisierung verfügt, antwortet er auf die HTTP-Bind-Anforderung mit der Statusaktualisierung als XMPP-Nachricht im HTTP-Antworttext.
4. Wenn 30 Sekunden nach Empfang der HTTP-Bind-Anforderung keine Statusaktualisierungen vorliegen, antwortet der Benachrichtigungsdienst ohne Statusaktualisierungen, damit der Finesse-Webclient eine weitere HTTP-Bind-Anforderung senden kann. Auf diese Weise kann der Benachrichtigungsdienst feststellen, dass der Finesse-Web-Client immer noch eine Verbindung zum

Benachrichtigungsdienst herstellen kann und dass der Agent den Browser nicht geschlossen oder den Computer nicht in den Ruhezustand versetzt hat usw.

Finesse XMPP-Nachrichten und XMPP-Knoten verstehen

Finesse implementiert auch die XMPP-Spezifikation XEP-0060: Publish-Subscribe. Der Zweck dieser Spezifikation ist es, dem XMPP-Server (Benachrichtigungsdienst) zu ermöglichen, Informationen abzurufen, die an XMPP-Knoten (Themen) veröffentlicht werden, und dann XMPP-Ereignisse an Entitäten zu senden, die für den Knoten abonniert sind. Im Fall von Finesse sendet der CTI-Server (Computer Telephony Integration) CTI-Nachrichten an den Finesse-Webservice, um Finesse über Konfigurationsaktualisierungen zu informieren, wie z. B. die Erstellung eines Agenten oder einer CSQ (Contact Service Queue) oder Informationen zu einem Anruf. Diese Informationen werden dann in eine XMPP-Nachricht umgewandelt, die der Finesse-Webdienst an den Finesse Notification-Dienst übermittelt. Der Finesse Notification Service sendet XMPP über BOSH-Nachrichten an Agenten, die für bestimmte XMPP-Knoten registriert sind.

Einige der im [Finesse Web Services Developer Guide](#) definierten Finesse API-Objekte sind XMPP-Knoten. Agenten- und Supervisor-Webclients von Finesse können Ereignisaktualisierungen für einige dieser XMPP-Knoten abonnieren, um aktuelle Informationen über Echtzeitergebnisse (z. B. Anrufereignisse, Zustandsereignisse usw.) zu erhalten. Diese Tabelle zeigt die XMPP-Knoten, die pubsub aktiviert sind.

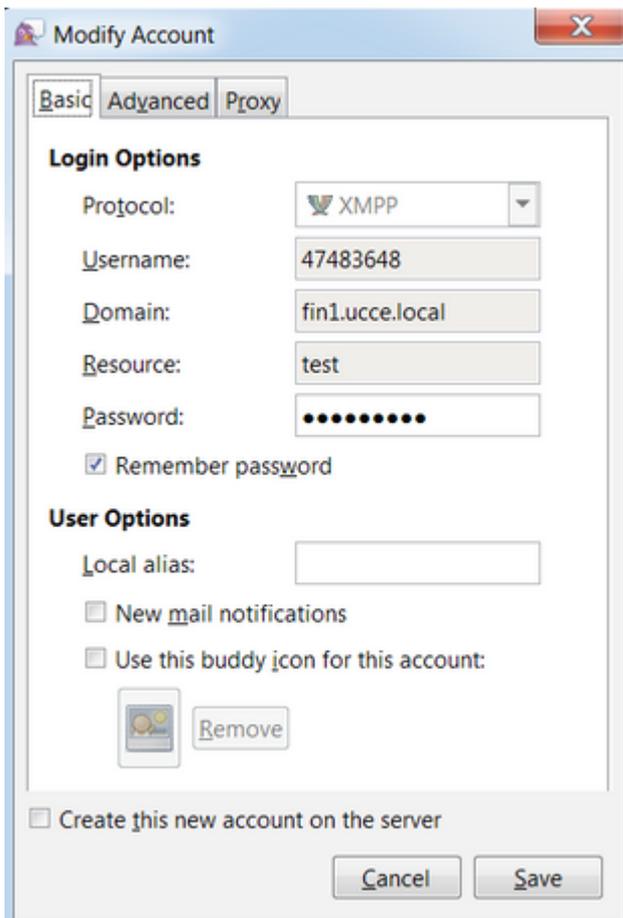
Finesse API-Objekt	Zweck	Abonnement
/finesse/api/User/<LoginID>	Zeigt den Status und die Teamzuordnung des Agenten an.	Agenten und Supervisoren
/finesse/api/User/<LoginID>/Dialogs	Zeigt die Anrufe an, die vom Mitarbeiter bearbeitet wurden.	Agenten und Supervisoren
/finesse/api/User/<LoginID>/ClientLog	Wird zum Erfassen von Client-Protokollen über die Schaltfläche Send Error Report (Fehlerbericht senden) verwendet.	Agenten und Supervisoren
/finesse/api/User/<LoginID>/Queue/<queueID>	Zeigt Warteschlangenstatistikdaten an (falls aktiviert)	Agenten und Supervisoren
/finesse/api/Team/<TeamID>/Benutzer	Zeigt die Agenten an, die zu einem bestimmten Team gehören, einschließlich Statusinformationen	Supervisoren
/finesse/api/SystemInfo	Zeigt den Status des Finesse-Servers an. Ermittelt, ob ein Failover erforderlich ist	Agenten und Supervisoren

Beispiel 1: Verwenden von Pidgin zum Anzeigen von Finesse XMPP-Knoten

Schritt 1: Laden Sie den XMPP-Client Pidgin herunter, und installieren Sie ihn.

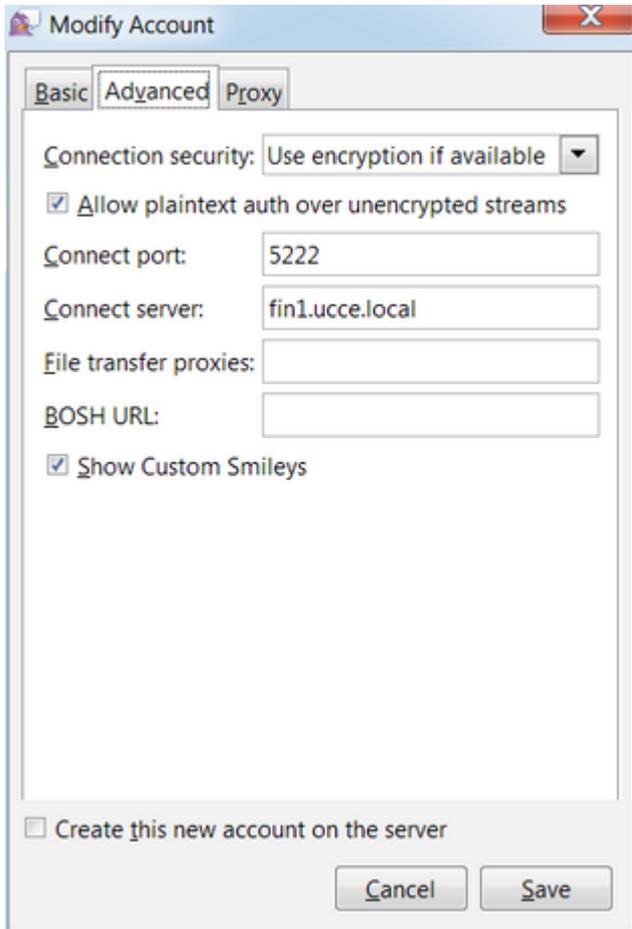
Schritt 2: Navigieren Sie zu **Konten > Ändern > Grundlegend**, und konfigurieren Sie die **Anmeldeoptionen**:

- Protokoll: XMPP
- Benutzername: Anmelde-ID für einen beliebigen Agenten
- Domäne: FQDN des Finesse-Servers
- Ressource: Platzhalter - jeder Wert kann verwendet werden, z. B. test
- Kennwort: Agent-Kennwort
- Aktivieren Sie das Kontrollkästchen **Kennwort speichern**.



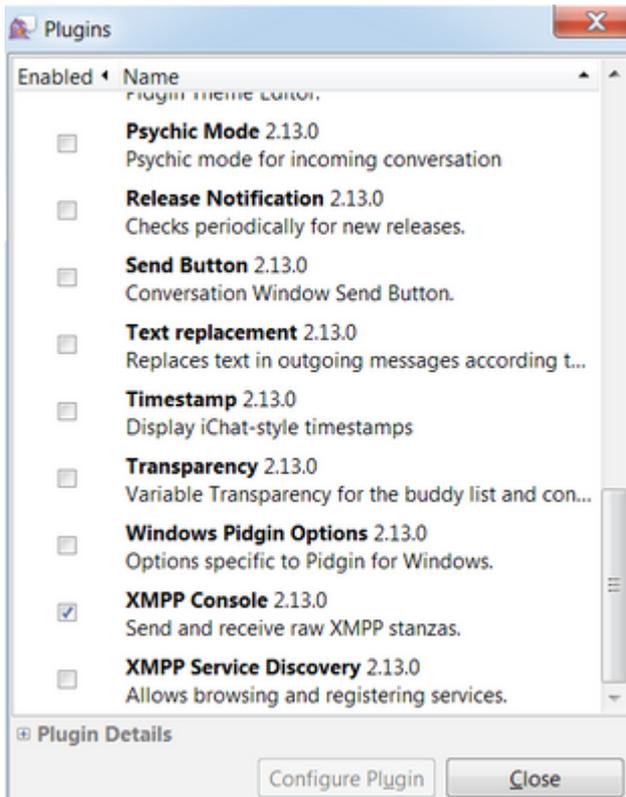
Schritt 3: Navigieren Sie zu **Konten > Ändern > Erweitert**, und konfigurieren Sie:

- Verbindungssicherheit: Falls verfügbar, Verschlüsselung verwenden
- Aktivieren Sie das Kontrollkästchen **Klartext zulassen, um andere unverschlüsselte Streams zu erstellen**.
- Verbindungsport: 5222. Verwenden Sie den Standardport 5222. Dieser Port ist für externe XMPP-Clients erforderlich. Finesse Desktop-Clients verwenden 7443. Verwenden Sie nicht den Port 7443.
- Verbindungsserver: Finesse-Server FQDN

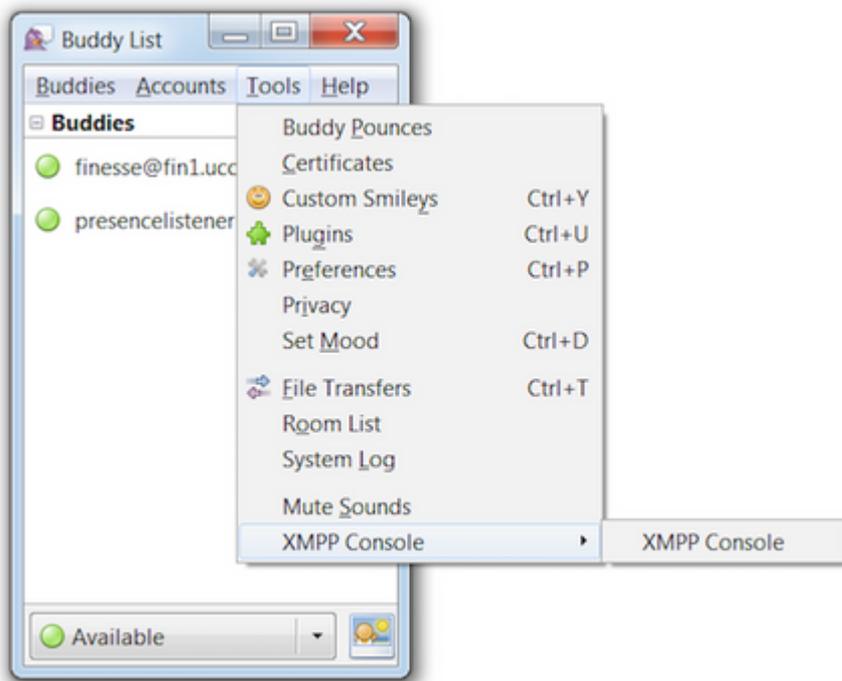


Hinweis: Port 5222 wird nur verwendet, weil Finesse-Web-Clients Port 7443 für die Verbindung mit dem Benachrichtigungsdienst verwenden können.

Schritt 4: Navigieren Sie zu **Tools > Plugins**, und aktivieren Sie die XMPP-Konsole.

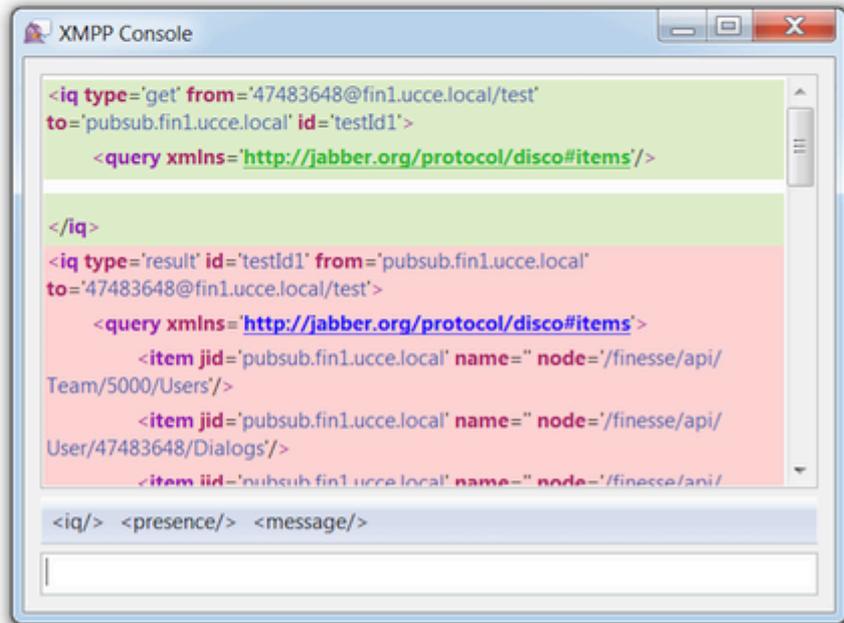
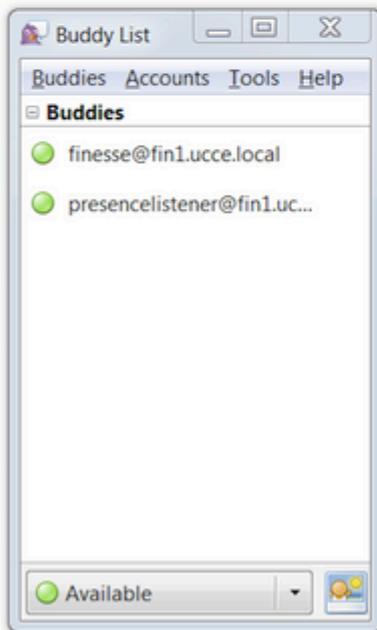
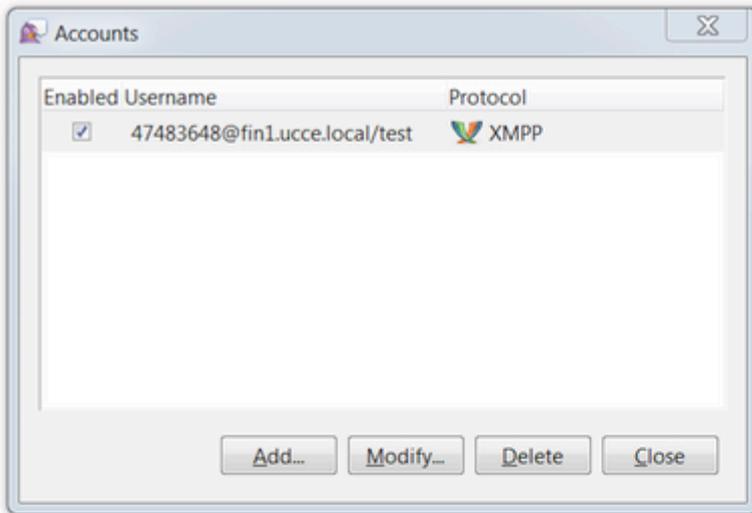


Schritt 5: Navigieren Sie zu **Extras > XMPP-Konsole > XMPP-Konsole**, um die XMPP-Konsole zu öffnen.



Schritt 6: Führen Sie diese **<iq>**-Meldung aus, um alle vorhandenen XMPP-Knoten anzuzeigen.

Beispiele:



In einer Laborumgebung mit zwei konfigurierten Agenten und zwei CSQs ist diese Ausgabe in der Finesse-Antwort enthalten:

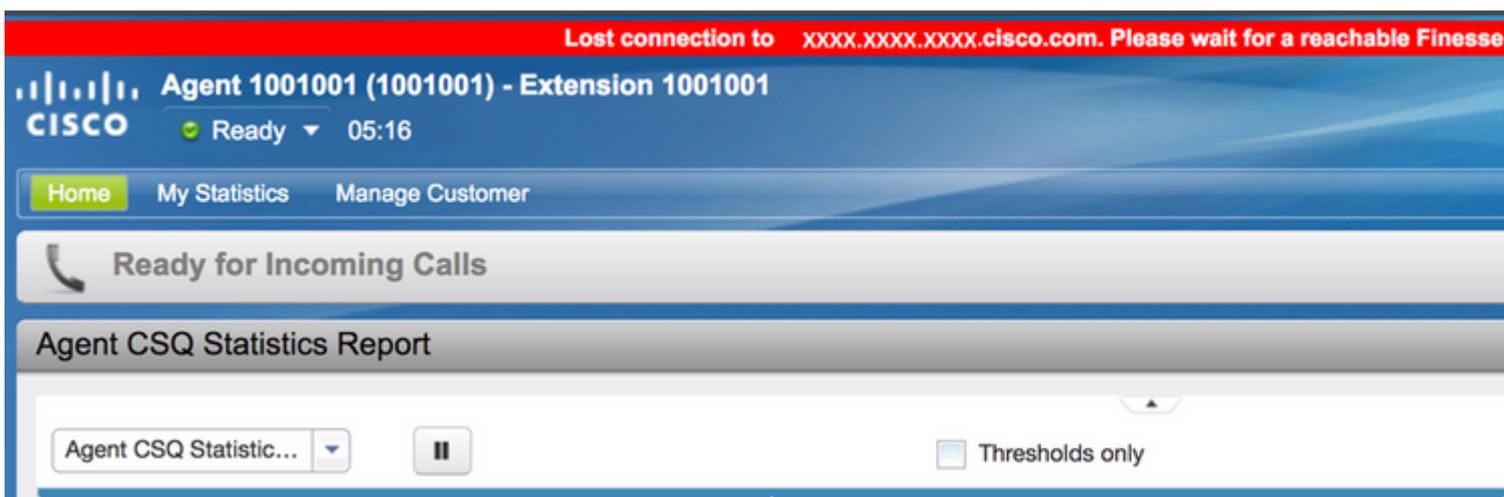
Beispiel 2: Verwenden der Registerkarte Netzwerk der Browser-Entwicklertools zum Anzeigen von HTTP-Nachrichten

Jeder Browser verfügt über eine Reihe von Entwicklungstools. Auf der Registerkarte "Netzwerk" der Entwicklungstools werden die vom Finesse-Webclient (Browser) gesendeten und empfangenen HTTP-Nachrichten angezeigt. Dieses Bild zeigt beispielsweise, wie der Finesse-Webclient eine SystemInfo-Anfrage sendet, die den Finesse-Tomcat-Status jede Minute als Failover-Prüfung überprüft. Zusätzlich werden auch die HTTP-Bind-Meldungen der BOSH-Verbindung angezeigt. Der Finesse-Server sendet innerhalb von 30 Sekunden eine Antwort zurück, wenn auf den XMPP-Knoten, für die der Web-Client abonniert ist, keine Updates zur Veröffentlichung vorhanden sind.

Status	Method	File	Domain	Cause	Type	Transfer...	Size	0 ms
200	POST	/http-bind/	XX.XX.XX.XX:7443	xhr	xml	57 B	57 B	
200	GET	Systeminfo?timestamponly&nocache=1492185680998	XX.XX.XX.XX:8445	xhr	xml	166 B	166 B	
200	POST	/http-bind/	XX.XX.XX.XX:7443	xhr	xml	57 B	57 B	
200	POST	/http-bind/	XX.XX.XX.XX:7443	xhr	xml	57 B	57 B	
200	GET	Systeminfo?timestamponly&nocache=1492185741004	XX.XX.XX.XX:8445	xhr	xml	166 B	166 B	
200	POST	/http-bind/	XX.XX.XX.XX:7443	xhr	xml	57 B	57 B	
200	POST	/http-bind/	XX.XX.XX.XX:7443	xhr	xml	57 B	57 B	
200	GET	Systeminfo?timestamponly&nocache=1492185801004	XX.XX.XX.XX:8445	xhr	xml	166 B	166 B	
200	POST	/http-bind/	XX.XX.XX.XX:7443	xhr	xml	57 B	57 B	
200	POST	/http-bind/	XX.XX.XX.XX:7443	xhr	xml	57 B	57 B	
200	GET	Systeminfo?timestamponly&nocache=1492185861006	XX.XX.XX.XX:8445	xhr	xml	166 B	166 B	
200	POST	/http-bind/	XX.XX.XX.XX:7443	xhr	xml	57 B	57 B	
200	POST	/http-bind/	XX.XX.XX.XX:7443	xhr	xml	57 B	57 B	

Fehlerbehebung: Fehlermeldung bei BOSH-Verbindungstrennung

Wenn eine BOSH-Verbindung getrennt wird, wird der Fehler Verbindung mit {Finesse Server FQDN} verloren. Bitte warten Sie, bis ein erreichbarer Finesse-Server gefunden wurde... wird in einem roten Banner oben auf dem Finesse-Desktop angezeigt.



Diese Meldung wird angezeigt, da derzeit keine XMPP-Abonnementereignisse vom Cisco Finesse Notification Service empfangen werden können. Daher können Statusinformationen und Anruferdetails nicht auf dem Mitarbeiter-Desktop angezeigt werden.

Bei UCCX wird der Agent 60 Sekunden nach dem Trennen der Browserverbindung in den Abmeldestatus versetzt. Der Agent kann sich im Status Ready (Bereit) oder Not Ready (Nicht Bereit) befinden, damit die Abmeldung erfolgt.

Für UCCE benötigt Finesse bis zu 120 Sekunden, um zu erkennen, wann ein Agent den Browser schließt oder der Browser abstürzt, und Finesse wartet 60 Sekunden, bevor eine erzwungene Abmeldeanforderung an den CTI-Server gesendet wird, die den CTI-Server veranlasst, den Agenten in den Zustand "Nicht bereit" zu versetzen. Unter diesen Bedingungen kann es bis zu 180 Sekunden dauern, bis Finesse den Agenten abmeldet. Im Gegensatz zu UCCX wechselt der Agent in den Status "Not Ready" (Nicht bereit) und nicht in den Status "Logout" (Abmelden).

Hinweis: CTI-Trennung nicht bereit vs. Das Abmeldestatusverhalten in UCCE wird durch den

Parameter PG /LOAD gesteuert. Gemäß den Versionshinweisen für Unified Contact Center Enterprise und Hosted Release 10.0(1) ist der Parameter /LOAD ab UCCE 10.0 veraltet.

Weitere Informationen zum Verhalten von UCCE Finesse Desktop finden Sie im Abschnitt "Desktop Behavior" des Kapitels "Cisco Finesse Failover Mechanism" im [Cisco Finesse Administration Guide](#).

Hinweis: Timer-Werte können sich je nach Produkthanforderung in der Zukunft ändern.

Protokollanalyse

Die Finesse- und UCCX-Benachrichtigungsdienstprotokolle können über RTMT oder die CLI erfasst werden:

file get activelog /desktop recurs compress

Debug Notification Service-Protokolle

Hinweis: Legen Sie Debug-Level-Protokolle nur fest, während Sie ein Problem reproduzieren. Deaktivieren Sie die Debugging-Funktion, nachdem das Problem reproduziert wurde.

Hinweis: Finesse 9.0(1) verfügt nicht über eine Protokollierung auf Debugebene. Die Protokollierung auf Debugebene wurde in Finesse 9.1(1) eingeführt. Der Prozess zum Aktivieren der Protokollierung unterscheidet sich in 9.1(1) von Finesse 10.0(1) - 11.6(1). Weitere Informationen hierzu finden Sie im Finesse Administrations- und Wartungsleitfaden.

Aktivieren Sie die Benachrichtigungs-Service-Debug-Protokolle von Unified Contact Center Express (UCCX) wie folgt:

```
<#root>
```

```
admin:
```

```
utils uccx notification-service log enable
```

```
WARNING! Enabling Cisco Unified CCX Notification Service logging can affect system performance and should be disabled when logging is not required.
```

```
Do you want to proceed (yes/no)? yes
```

```
Cisco Unified CCX Notification Service logging enabled successfully.
```

```
NOTE: Logging can be disabled automatically if Cisco Unified CCX Notification Service is restarted.
```

Aktivieren Sie die Benachrichtigungs-Service-Debug-Protokolle von Unified Contact Center Enterprise (UCCE) (Finesse Standalone) wie folgt:

```
<#root>
```

admin:

utils finesse notification logging enable

Checking that the Cisco Finesse Notification Service is started...
The Cisco Finesse Notification Service is started.

Cisco Finesse Notification Service logging is now enabled.

WARNING! Cisco Finesse Notification Service logging can affect system performance and should be disabled when logging is not required.

Note: Logging can be disabled automatically if you restart the Cisco Finesse Notification Service

Diese Protokolle befinden sich im Ordner /desktop/logs/openfire und haben den Namen debug.log.

Wie im Bild gezeigt, zeigt der Benachrichtigungsdienst (Openfire) debug.log die HTTP-Bindung mit Desktop sowie die IP-Adresse und den Port des Agent-PCs.

```
XXX.XXX.XXX.XX:1:34:21 [Session-1, SSL_NULL_WITH_NULL_NULL] received 0 sent 0
2017.04.14 21:34:21 REQUEST /http-bind/ on org.eclipse.jetty.server.nio.SelectChannelConnector$SelectChannelHttpConnection@2d5a26@XXX.XXX.XXX.XXX
2017.04.14 21:34:21 scope null|/http-bind/ @ o.e.j.s.ServletContextHandler{/http-bind,null}
2017.04.14 21:34:21 context=/http-bind|/ @ o.e.j.s.ServletContextHandler{/http-bind,null}
2017.04.14 21:34:21 sessionManager=org.eclipse.jetty.server.session.HashSessionManager@176fe4#STARTED
2017.04.14 21:34:21 session=null
2017.04.14 21:34:21 session=null
2017.04.14 21:34:21 servlet /http-bind|/ -> org.jivesoftware.openfire.http.HttpBindServlet-1643193
2017.04.14 21:34:21 chain=null
2017.04.14 21:34:21 HTTPBindLog: HTTP RECV(3445afbe): <body sid="3445afbe" rid="164053266"/>
2017.04.14 21:34:21 consumeResponse: org.jivesoftware.openfire.http.HttpSession@dd7653 status: 3 address: 1001003@XXX.XXXX.XXX.XXX.XX.cisco.com
<presence from="1001003@XXX.XXXX.XXX.XXX.XX.cisco.com/desktop">
  <c xmlns="http://jabber.org/protocol/caps" hash="sha-1" node="http://jabber.cisco.com/cax1" ver="VNC6fNwvCxe6FJfDJIpLryVJRwM="/>
</presence> rid: 164053266
2017.04.14 21:34:21 suspended org.eclipse.jetty.server.nio.SelectChannelConnector$SelectChannelHttpConnection@2d5a26@XXX.XXX.XXX.XX:7443<->
2017.04.14 21:34:24 Launching thread for /127.0.0.1:44667
2017.04.14 21:34:24 Launching thread for /127.0.0.1:44656
```

Wie im Bild gezeigt, zeigt die letzte aktive 0 ms, dass die Sitzung noch aktiv ist.

```
2017.04.14 21:34:26 Exiting since queue is empty for /127.0.0.1:44660
2017.04.14 21:34:26 Session (id=3445afbe) was last active 0 ms ago: 1001003@XXXXXXXXX.XXXXXXXXXX.cisco.com/
2017.04.14 21:34:26 time=1492185866851,JID=1001003@XXXXXXXXX.XXXXXXXXXX.cisco.com/desktop,msgs_sent=4,msgs_
2017.04.14 21:34:26 time=1492185866851,JID=1001003@XXXXXXXXX.XXXXXXXXXX.cisco.com/desktop,msgs_sent=4,msgs_
```

Das Schließen der inaktiven Sitzung durch OpenFire zeigt an, dass die Agenten-Abmeldung innerhalb von 60 Sekunden ausgelöst werden kann, wobei Finesse eine erzwungene Abmeldung mit dem Ursachencode 255 an den CTI-Server senden kann. Das tatsächliche Verhalten des Desktops unter diesen Bedingungen hängt von der Einstellung für "Logout on Agent Disconnect (LOAD)" in UCCE ab. In UCCX ist dies immer das Verhalten.

Wenn der Finesse-Client keine HTTP-Bind-Meldungen an den Finesse-Server sendet, können die Protokolle die Sitzungslaufzeit und das Schließen der Sitzung anzeigen.

```
2017.06.17 00:14:34 Session (id=f382a015) was last active 0 ms ago: 1001003@xxxxx.xxxx.xxx.cisco.com/des
2017.06.17 00:15:04 Session (id=f382a015) was last active 13230 ms ago: 1001003@xxxxx.xxxx.xxx.cisco.com
2017.06.17 00:15:34 Session (id=f382a015) was last active 43230 ms ago: 1001003@xxxxx.xxxx.xxx.cisco.com
2017.06.17 00:16:04 Session (id=f382a015) was last active 63231 ms ago: 1001003@xxxxx.xxxx.xxx.cisco.com
2017.06.17 00:17:04 Unable to route packet. No session is available so store offline. <message from="pub
```

Info Notification Service-Protokolle

Diese Protokolle befinden sich im Ordner /desktop/logs/openfire und haben den Namen info.log. Wenn der Finesse-Client keine HTTP-Bind-Meldungen an den Finesse-Server sendet, können die Protokolle anzeigen, dass die Sitzung inaktiv wird.

```
2017.06.17 00:16:04 Closing idle session (id=f382a015): 1001003@xxxxx.xxxx.xxx. cisco.com/desktop
after inactivity for more than threshold value of 60
2017.06.17 00:16:04 A session is closed for 1001003@xxxxx.xxxx.xxx. cisco.com/desktop
```

Webservices-Protokolle

Diese Protokolle befinden sich im Ordner /desktop/logs/webservices und haben den Namen Desktop-webservices.YYYY-MM-DDTHH-MM-SS.sss.log. Wenn der Finesse-Client innerhalb der festgelegten Zeit keine HTTP-Bind-Meldungen an den Finesse-Server sendet, kann aus den Protokollen hervorgehen, dass der Agent nicht mehr verfügbar ist, und 60 Sekunden später kann es zu einem präsenzgesteuerten Logout kommen.

```
0000001043: XX.XX.XX.XXX: Jun 17 2017 00:16:04.630 +0530: %CCBU_Smack Listener Processor (1)-6-PRESENCE
0000000417: XX.XX.XX.XXX: Jun 17 2017 00:16:04.631 +0530: %CCBU_Smack Listener Processor (1)-6-UNSUBSCRIBED
0000001044: XX.XX.XX.XXX: Jun 17 2017 00:16:04.631 +0530: %CCBU_Smack Listener Processor (1)-6-AGENT_PRESENCE
0000001051: XX.XX.XX.XXX: Jun 17 2017 00:16:35.384 +0530: %CCBU_pool-8-thread-1-6-AGENT_PRESENCE_MONITORING
0000001060: XX.XX.XX.XXX: Jun 17 2017 00:17:04.632 +0530: %CCBU_CoreImpl-worker12-6-PRESENCE DRIVEN LOGOUT
0000001061: XX.XX.XX.XXX: Jun 17 2017 00:17:04.633 +0530: %CCBU_CoreImpl-worker12-6-MESSAGE_TO_CTI_SERVER
1, workmode : 0, reason code: 255, forceflag :1, agentcapacity: 1, agenttext: 1001003, agentid: 1001003,
0000001066: XX.XX.XX.XXX: Jun 17 2017 00:17:04.643 +0530: %CCBU_CTIMessageEventExecutor-0-6-DECODED_MESSAGE
skillGroupNumber=-1, skillGroupPriority=0, agentState=1 (LOGOUT), eventReasonCode=255, numFltSkillGroups=0,
duration=null, nextAgentState=null, fltSkillGroupNumberList=[], fltSkillGroupIDList=[], fltSkillGroupPriorityList=[],
msgID=30, timeTracker={"id":"AgentStateEvent","CTI_MSG_RECEIVED":1497638824642,"CTI_MSG_DISPATCH":1497638824642}
Decoded Message to Finesse from backend cti server
```

Allgemeine Gründe für BOSH-Trennung

BOSH-Verbindungen werden vom Web-Client eingerichtet, und der Finesse-Server bestimmt, ob der Agent-Presence-Status nicht verfügbar ist. Bei diesen Problemen handelt es sich fast immer um clientseitige Probleme in Bezug auf den Browser, den Agentencomputer oder das Netzwerk, da der Verbindungsstart vom Client abhängt.

Problem - Verbindung der Agenten zu verschiedenen Zeiten getrennt (Client-seitiges Problem)

Empfohlene Maßnahmen

Überprüfen Sie, ob folgende Probleme vorliegen:

1. Netzwerkproblem:

- Firewall-Regeln und -Protokolle überprüfen - TCP-Port 7443 darf nicht blockiert oder gedrosselt werden
- Verwenden Sie einen HTTP-Web-Traffic-Sniffer wie [Fiddler®](#) oder [Wireshark®](#), um zu bestätigen, dass der Browser HTTP-Bind-Anfragen über den TCP-Port 7443 sendet und Antworten empfängt.
- Überprüfen Sie alle Netzwerkgeräte/Schnittstellen zwischen dem Agent-Computer und dem Finesse-Server auf übermäßige Verzögerungen oder Paketverluste.
 - Traceroute kann zur Ermittlung des Pfads und von Verzögerungen nützlich sein.
 - Auf einem Microsoft® Windows® PC: tracert {Finesse Server IP | Finesse Server-FQDN}
 - Auf einem Mac®: Traceroute {Finesse Server IP | Finesse Server-FQDN}
 - In der Cisco IOS® Software können die Schnittstellenstatistiken überprüft werden: Schnittstellen anzeigen
 - Weitere Informationen finden Sie [unter Fehlerbehebung bei Einbrüchen von Eingabewarteschlangen und Einbrüchen von Ausgabewarteschlangen](#)
- Sammeln von Finesse-Client-Protokollen für einen Test-Agenten Client-Protokolle können auf drei Arten erfasst werden:
 1. Webkonsolen-Protokolle für Browser
 - [Firefox-Webkonsole](#)
 - [Microsoft Edge-Webkonsole](#)
 - [Chrome-Webkonsole](#)
 2. Drücken Sie auf der Seite Finesse die Schaltfläche [Send Error Report \(Fehlerbericht senden\)](#), und sammeln Sie die Finesse-Serverprotokolle. Die Protokolle befinden sich unter /desktop/logs/clientlogs.
 3. Melden Sie sich über <https://<Finesse-FQDN>/desktop/localLog> an, und erfassen Sie die Protokolle, nachdem das Problem aufgetreten ist.

Der Client stellt jede Minute eine Verbindung zum Finesse-Server her, um die Drift und die Netzwerklatenz zu berechnen:

```
<PC date-time with GMT offset>: : <Finesse FQDN>: <Finesse server date-time with offset>:
Header : Client: <date-time>, Server: <date-time>, Drift: <drift> ms, Network Latency (round trip): <RTT>
2019-01-11T12:24:14.586 -05:00: : fin1.ucce.local: Jan 11 2019 11:24:14.577 -0600: Header : Client: 2019
```

Bei Protokollsammelungsproblemen finden Sie weitere Informationen unter [Problembehandlung bei Cisco Finesse Desktop Persistent Logging Problem](#)

2. Nicht unterstützter Browser und/oder Version:

Verwenden Sie die unterstützten Browser/Versionen und Einstellungen gemäß den Kompatibilitätsmatrizen:

[UCCE-Kompatibilitätsmatrix](#)

[UCCX-Kompatibilitätsmatrix](#)

3. Der Zustand des Browsers blieb aufgrund des Inhalts/der Verarbeitung anderer Registerkarten/Fenster hängen:

Überprüfen Sie den Agenten-Workflow, um festzustellen, ob sie:

- In der Regel gibt es Registerkarten oder Fenster, auf denen ständig andere Echtzeitanwendungen ausgeführt werden, z. B. Musik-/Video-Streaming, WebSocket-Verbindungen, benutzerdefinierte CRM-Webclients usw.
- Sehr viele Registerkarten oder Fenster geöffnet haben
- Browser-Caching deaktiviert haben
- Ihren Browser über einen langen Zeitraum laufen lassen und den Browser nicht am Ende des Arbeitstages schließen

4. Computer in den Schlaf versetzt:

Überprüfen Sie, ob der Agent den Computer in den Energiesparmodus versetzt, bevor Sie sich von Finesse abmelden, oder ob der Zeitgeber für die Einstellung des Energiesparmodus sehr niedrig ist.

5. Hoher CPU- oder hoher Speicherbedarf auf dem Client-Computer:

- Wenn der Agent-Browser in einer freigegebenen Umgebung wie Microsoft Windows Remote Desktop Services, Citrix® XenApp® oder Citrix XenDesktop® ausgeführt wird, stellen Sie fest, ob die Leistung des Browsers von der Anzahl der Benutzer abhängt, die den Browser gleichzeitig ausführen.
 - Stellen Sie sicher, dass der richtige Arbeitsspeicher und die richtigen CPU-Ressourcen auf Basis der Anzahl der Benutzer konfiguriert werden.
- Probleme bei der Nutzung der Computerressourcen überprüfen:
 - Windows:
 - Windows [PowerShell Get-Counter](#)-Befehl, der % der CPU-Zeit, MB verfügbaren Arbeitsspeichers und % des verwendeten Arbeitsspeichers alle 2 Sekunden überprüft: `Get-Counter -Counter "\Processor(_Total)\% Processor Time", "\Memory\Available MBytes", "\Memory\% Committed Bytes In Use" -SampleInterval 2 -Continuous`
 - Als Alternative zur Verwendung von PowerShell zum Anzeigen der Windows-Leistungsindikatoren kann [Windows Performance Monitor](#) verwendet werden.
 - [Der Task-Manager](#) kann verwendet werden, um Live-CPU- und Speicherstatistiken global und auf Prozessbasis anzuzeigen.
 - Mac:
 - [Terminal Top](#)-Befehl, der die Live-Gesamt-CPU und den Arbeitsspeicher überprüft: `top`
 - Prozesse überprüfen und nach CPU-Auslastung sortieren: `top -o CPU`
 - Prozesse überprüfen und nach Speichernutzung sortieren: `top -o MEM`
 - [Activity Monitor](#) kann verwendet werden, um Live-CPU- und Arbeitsspeicherstatistiken global und auf Prozessbasis anzuzeigen

6. Gadgets von Drittanbietern, die unerwartete, problematische Aktivitäten im Hintergrund ausführen:

Testen Sie das Verhalten des Finesse-Desktops, wenn alle Gadgets von Drittanbietern entfernt wurden.

7. NTP-Problem auf Server oder Client:

- Überprüfen Sie **utils ntp status** auf dem Finesse Publisher-Server, um sicherzustellen, dass die NTP-Serverschicht 4 oder niedriger ist.
- Überprüfung der Drift und Netzwerklatenz in den Client-Protokollen

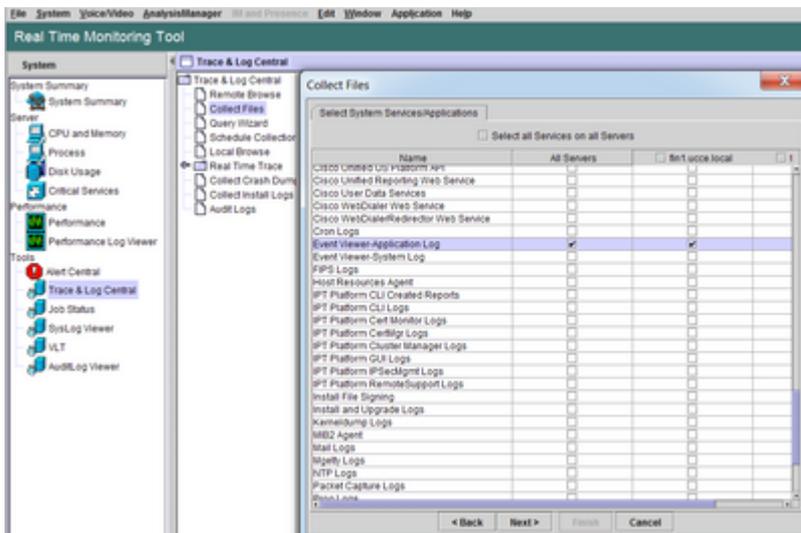
Problem - Alle Agenten trennen gleichzeitig die Verbindung (serverseitiges Problem)

Empfohlene Maßnahmen

Überprüfen Sie, ob folgende Probleme vorliegen:

1. Cisco Unified Communications Manager CTIManager-Dienst wird getrennt. Wenn alle CTIManager-Anbieter für UCCX heruntergefahren sind oder abstürzen, wird den UCCX-Agenten der rote Bannerfehler angezeigt. In diesem Fall wird den UCCE-Agenten das rote Banner nicht angezeigt, aber Anrufe werden nicht richtig an die Agenten weitergeleitet.

- Überprüfen Sie, ob der Cisco CTIManager-Dienst auf den CUCM-Servern gestartet wird, die als CTI-Anbieter verwendet werden.
- Überprüfen Sie, ob der Cisco CTIManager-Dienst über die Ereignisanzeige abgestürzt ist. Die Anwendung meldet sich bei RTMT an, um zu sehen, ob der Cisco CTIManager-Dienst abgestürzt ist.
 - Um Ereignisanzeige-Protokolle auf RTMT zu sammeln, navigieren Sie zu **System > Tools > Trace and Log Central > Collect Files > Select System Services/Applications > Event Viewer-Application Log**.



- Um die Ereignisanzeige-Anwendungsprotokolle in CLI zu sammeln, rufen Sie die Datei `activelog /syslog/CiscoSyslog*` abtime `hh:mm:MM/TT/JJ hh:mm:MM/TT/JJ` auf.
- So zeigen Sie Core Dumps in der CLI an: `utils core active list`

Hinweis: Core Dumps-Dateinamen verwenden das Format `core.<ProcessID>.<SignalNumber>.<ProcessName>.<EpochTime>`.
Beispiel: `core.24587.6.CTIManager.1533441238`
Somit kann aus der Epochenzeit der Zeitpunkt des Crashes ermittelt werden.

2. Finesse/UCCX-Benachrichtigungsdienst wurde beendet oder ist abgestürzt:

- Überprüfen Sie die Ereignisanzeige-Anwendungsprotokolle auf Fehler des Benachrichtigungsdiensts, oder stellen Sie fest, ob der Dienst beendet wurde.
- Überprüfen Sie, ob der Benachrichtigungsdienst aktiv ist: Dienstliste "utils"
- Überprüfen Sie, wann der Benachrichtigungsdienst heruntergefahren wurde: `file search active log /desktop/logs/openfire "Openfire stop"`
- Überprüfen Sie die Startzeiten des Benachrichtigungsdiensts: `file search active log /desktop/logs/openfire "HTTP bind service started"` (HTTP-Bindungsdienst gestartet).
- Suchen Sie nach Speicherabbildern des Benachrichtigungsdiensts, die nach einem Absturz entstanden sind: Dateiliste `activelog /desktop/logs/openfire/*.hprof`
- Überprüfen Sie, ob der Benachrichtigungsdienst auf Datenverkehr auf dem TCP-Port 7443 wartet: `show open ports regexp 7443.*LISTEN`
- Prüfen Sie, ob diese Fehler zutreffen (diese Fehler würden bei angemeldeten Agenten zu einem

Anmeldefehler führen und bei bereits angemeldeten Agenten würde für diese Agenten die rote Bannernachricht Finesse disconnect angezeigt):

- Cisco Bug-ID [CSCva7280](#) - Finesse Tomcat und Openfire Crash für ungültige XML-Zeichen
- Cisco Bug-ID [CSCva7235](#) - UCCX: Finesse Tomcat und Openfire-Absturz wegen ungültiger XML-Zeichen

Starten Sie Cisco Finesse Tomcat und Notification Service neu, wenn ein Absturz vermutet wird. Dies wird nur bei einem Netzwerkausfall empfohlen. Andernfalls werden die Agenten bei einem Neustart vom Finesse-Server getrennt.

Schritte für UCCE:

- `utils service stop Cisco Finesse Tomcat`
- `utils service stop Cisco Finesse Notification Service`
- `utils service start Cisco Finesse Tomcat`
- `utils service start Cisco Finesse Notification Service`

Schritte für UCCX:

- `utils service stop Cisco Finesse Tomcat`
- `utils service stop Cisco Unified CCX Notification Service`
- `utils service start Cisco Finesse Tomcat`
- `utils service start Cisco Unified CCX Notification Service`

Fiddler verwenden

Die Konfiguration von Fiddler kann eine ziemlich schwierige Aufgabe sein, ohne die erforderlichen Schritte zu verstehen und zu verstehen, wie Fiddler funktioniert. Fiddler ist ein Man-in-the-Middle-Webproxy, der zwischen dem Finesse-Client (Webbrowser) und dem Finesse-Server steht. Aufgrund der gesicherten Verbindungen zwischen dem Finesse-Client und dem Finesse-Server erhöht sich die Komplexität der Fiddler-Konfiguration, um gesicherte Nachrichten anzuzeigen.

Häufiges Fiddler-Problem

Da Fiddler zwischen dem Finesse-Client und dem Finesse-Server steht, muss die Fiddler-Anwendung signierte Zertifikate für alle Finesse TCP-Ports erstellen, die Zertifikate erfordern:

Cisco Finesse Tomcat Service-Zertifikate

1. Finesse Publisher-Server TCP 8445 (und/oder 443 für UCCE)
2. Finesse Subscriber-Server TCP 8445 (und/oder 443 für UCCE)

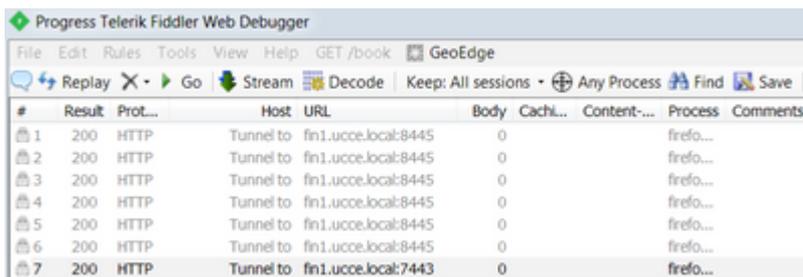
Cisco Finesse (Unified CCX) Notification Service-Zertifikate

1. Finesse Publisher-Server TCP 7443
2. Finesse Subscriber-Server TCP 7443

Die HTTPS-Entschlüsselung muss aktiviert sein, damit Fiddler dynamisch Zertifikate für den Finesse-Server generieren kann. Diese Funktion ist nicht standardmäßig aktiviert.

Wenn die HTTPS-Entschlüsselung nicht konfiguriert ist, wird die ursprüngliche Tunnelverbindung zum Benachrichtigungsdienst angezeigt, der HTTP-Bind-Datenverkehr jedoch nicht. Fiddler zeigt nur:

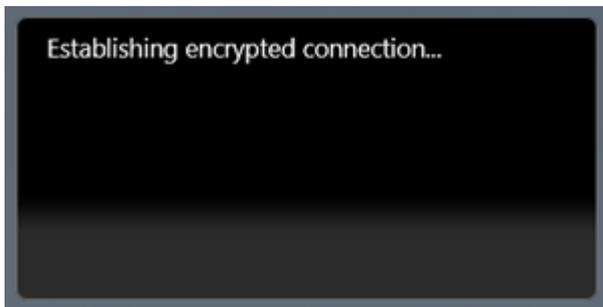
Tunnel to <Finesse server FQDN>:7443



The screenshot shows the Fiddler Web Debugger interface. The title bar reads 'Progress Telerik Fiddler Web Debugger'. The menu bar includes 'File', 'Edit', 'Rules', 'Tools', 'View', 'Help', and 'GET /book'. The toolbar contains 'Replay', 'Go', 'Stream', 'Decode', 'Keep: All sessions', 'Any Process', 'Find', and 'Save'. Below the toolbar is a table with the following columns: '#', 'Result', 'Prot...', 'Host', 'URL', 'Body', 'Cachi...', 'Content...', 'Process', and 'Comments'. The table contains seven rows of data, all with a status of '200' and 'HTTP'. The 'Host' column for the first six rows is 'Tunnel to fin1.uccelocal:8445', and for the seventh row, it is 'Tunnel to fin1.uccelocal:7443'. The 'Process' column for all rows shows 'frefo...'. The 'Body' and 'Cachi...' columns are empty.

#	Result	Prot...	Host	URL	Body	Cachi...	Content...	Process	Comments
1	200	HTTP	Tunnel to	fin1.uccelocal:8445	0			frefo...	
2	200	HTTP	Tunnel to	fin1.uccelocal:8445	0			frefo...	
3	200	HTTP	Tunnel to	fin1.uccelocal:8445	0			frefo...	
4	200	HTTP	Tunnel to	fin1.uccelocal:8445	0			frefo...	
5	200	HTTP	Tunnel to	fin1.uccelocal:8445	0			frefo...	
6	200	HTTP	Tunnel to	fin1.uccelocal:8445	0			frefo...	
7	200	HTTP	Tunnel to	fin1.uccelocal:7443	0			frefo...	

Anschließend müssen die von Fiddler signierten Finesse-Zertifikate vom Client als vertrauenswürdig eingestuft werden. Wenn diese Zertifikate nicht vertrauenswürdig sind, ist ein Übergang über die Phase der Herstellung der verschlüsselten Verbindung... der Finesse-Anmeldung nicht möglich.



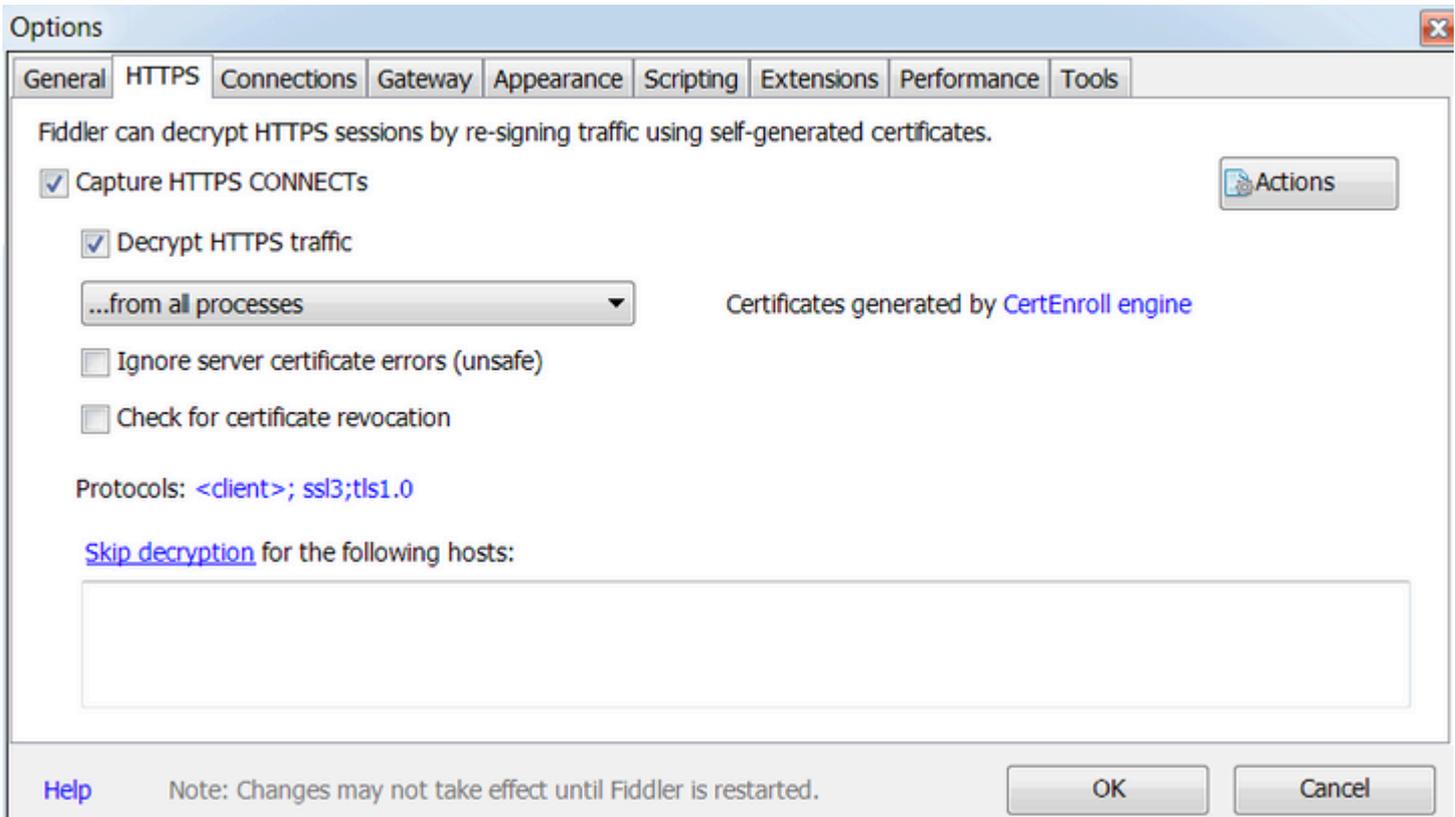
In einigen Fällen funktioniert das Akzeptieren der Zertifikatausnahmen von der Anmeldung nicht, und die Zertifikate müssen vom Browser manuell als vertrauenswürdig eingestuft werden.

Beispiel für Konfigurationsschritte

Achtung: Die Beispielkonfiguration ist für Fiddler v5.0.20182.28034 für .NET 4.5 und Mozilla Firefox 64.0.2 (32-Bit) unter Windows 7 x64 in einer Laborumgebung vorgesehen. Diese Verfahren können nicht für alle Versionen von Fiddler, alle Browser oder alle Computer-Betriebssysteme verallgemeinern. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die potenziellen Auswirkungen jeder Konfiguration kennen. Weitere Informationen finden Sie in der [offiziellen Fiddler-Dokumentation](#).

Schritt 1: Fiddler herunterladen

Schritt 2: HTTPS-Entschlüsselung aktivieren. Navigieren Sie zu **Extras > Optionen > HTTPS**, und aktivieren Sie das Kontrollkästchen **HTTPS-Datenverkehr entschlüsseln**.

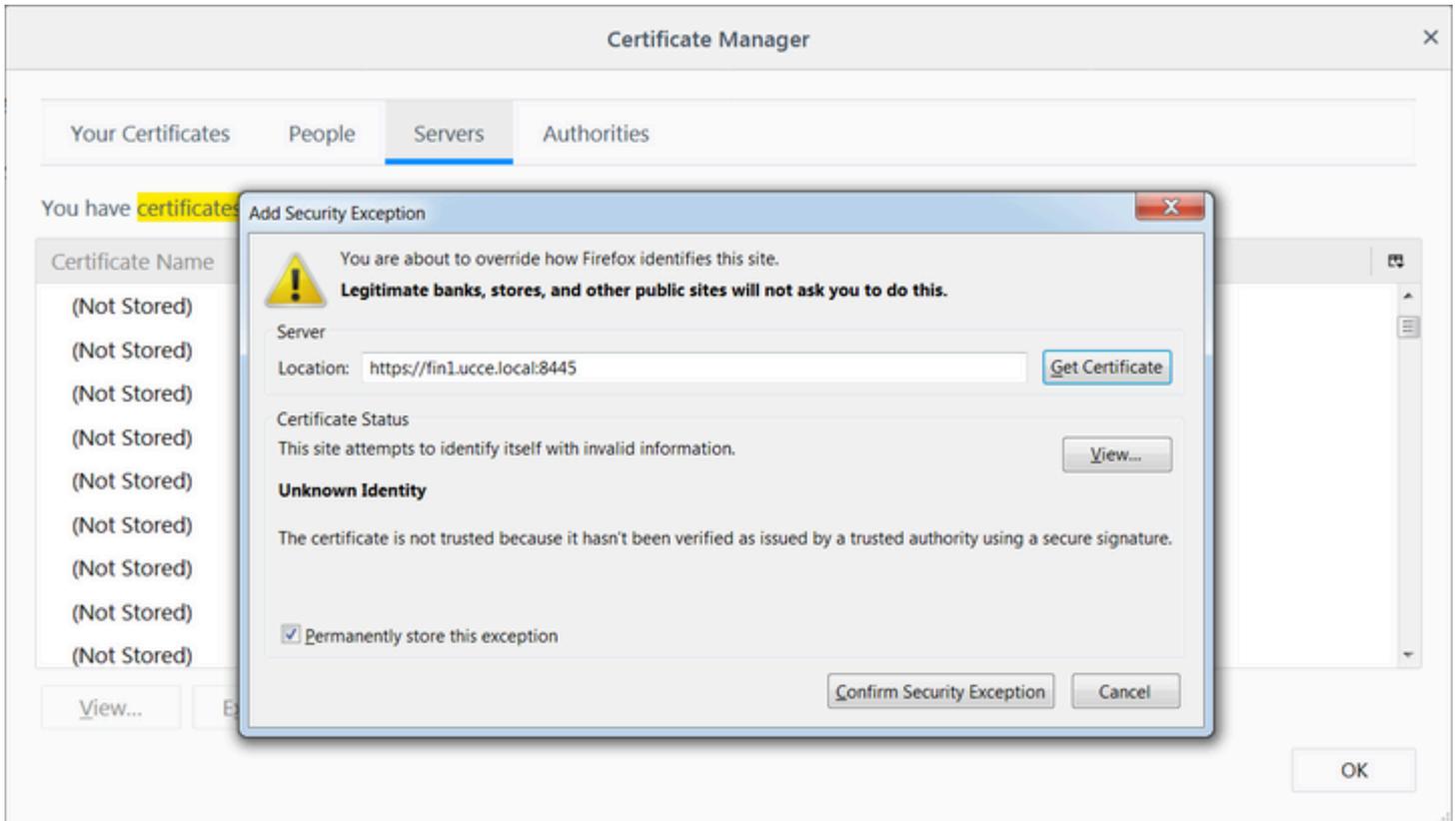


Schritt 3: Ein Warnmeldungsfield wird geöffnet, in dem Sie gefragt werden, ob Sie dem Fiddler-Stammzertifikat vertrauen. Wählen Sie **Ja aus**.

Schritt 4: Ein Warnmeldungsfield mit der Meldung "Sie sind im Begriff, ein Zertifikat von einer Zertifizierungsstelle (CA) zu installieren, die behauptet, Folgendes darzustellen: DO_NOT_TRUST_FiddlerRoot... Möchten Sie dieses Zertifikat installieren?" Wählen Sie **Ja aus**.

Schritt 5: Fügen Sie die Finesse Publisher- und Subscriber-Zertifikate manuell zum Zertifikatvertrauensspeicher des Computers oder Browsers hinzu. Stellen Sie die Ports 8445, 7443 und (nur für UCCE) 443 sicher. Auf Firefox kann dies beispielsweise einfach durchgeführt werden, ohne Zertifikate von der Seite "Finesse Operating System Administration" herunterzuladen:

Optionen > Suchen unter Optionen (Suche) > Zertifikate > Server > Ausnahme hinzufügen > Standort > Geben Sie https://<Finesse-Server>:Port für die entsprechenden Ports für beide Finesse-Server ein.



Schritt 6: Melden Sie sich bei Finesse an, und sehen Sie sich die http-bind-Meldungen an, die den Finesse-Client über Fiddler dem Finesse-Server überlassen.

Im angegebenen Beispiel zeigen die ersten 5 Nachrichten HTTP-Bind-Nachrichten, die vom Finesse-Server beantwortet wurden. Die erste Nachricht enthält 1571 Byte Daten, die im Nachrichtentext zurückgegeben werden. Der Text enthält ein XMPP-Update für ein Agent-Ereignis. Die letzte HTTP-Bind-Nachricht wurde vom Finesse-Client gesendet, hat jedoch keine Antwort vom Finesse-Server erhalten. Dies kann bestimmt werden, wenn das HTTP-Ergebnis NULL (-) und die Anzahl der Bytes im Antworttext NULL (-1) ist.

Progress Telerik Fiddler Web Debugger

File Edit Rules Tools View Help GET /book GeoEdge

Replay X Go Stream Decode Keep: All sessions Any Process Find Save Browse Clear Cache TextWizard Tearoff

#	Result	Prot...	Host	URL	Body	Cach...	Content...	Process	Comments	Custo
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	1,135		text/java...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	1,655		text/java...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	3,579		text/java...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	4,744		text/java...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	1,630		text/java...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	812		text/html	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	729		text/html	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	352		text/html	firefo...		
6...	200	HTTP	detectportal.fire...	/success.txt	8	no-ca...	text/plain	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	244		text/html	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	731		text/html	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	901		text/html	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	1,302		text/html	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	307		text/html	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	287		text/html	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	569		text/html	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	910		text/html	firefo...		
6...	200	HTTP	detectportal.fire...	/success.txt	8	no-ca...	text/plain	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	43		image/gif	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/ciscowidge...	1,176		text/html	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/theme/fine...	673		image/gif	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/ciscowidge...	720		text/html	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/finesse/api/User/47...	631	no-ca...	applicato...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/thirdparty/...	12,7...		image/png	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/theme/fine...	2,205		image/png	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/finesse/api/User/47...	340	no-ca...	applicato...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/finesse/api/User/47...	1,851	no-ca...	applicato...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/finesse/api/User/47...	20	no-ca...	applicato...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/gadgets/makeRequ...	340	no-ca...	applicato...	firefo...		
6...	200	HTTP	Tunnel to	cuic1.uacce.local:8444	0		firefo...			
6...	200	HTTPS	fin1.uacce.local...	/gadgets/makeRequ...	340	no-ca...	applicato...	firefo...		
6...	200	HTTP	detectportal.fire...	/success.txt	8	no-ca...	text/plain	firefo...		
6...	200	HTTP	Tunnel to	cuic1.uacce.local:8444	0		firefo...			
6...	200	HTTP	detectportal.fire...	/success.txt	8	no-ca...	text/plain	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/http-bind/	1,571		text/xml...	firefo...		
6...	202	HTTPS	fin1.uacce.local...	/finesse/api/User/47...	0	no-ca...	applicato...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/desktop/theme/fine...	673		image/gif	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/http-bind/	57		text/xml...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/finesse/api/SystemL...	232	no-ca...	applicato...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/http-bind/	57		text/xml...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/http-bind/	57		text/xml...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/finesse/api/SystemL...	232	no-ca...	applicato...	firefo...		
6...	200	HTTPS	fin1.uacce.local...	/http-bind/	57		text/xml...	firefo...		
6...	-	HTTPS	fin1.uacce.local...	/http-bind/	-1		firefo...			
6...	200	HTTPS	fin1.uacce.local...	/finesse/api/SystemL...	232	no-ca...	applicato...	firefo...		

Statistics Inspectors AutoResponder Compos

Headers TextView SyntaxView WebForms HexView

POST https://fin1.uacce.local:7443/http-bind/ HT
 Host: fin1.uacce.local:7443
 User-Agent: Mozilla/5.0 (windows NT 6.1; WOW64;
 Accept: text/plain, */*; q=0.01
 Accept-Language: en-US,en;q=0.5
 Accept-Encoding: gzip, deflate, br
 Referer: https://fin1.uacce.local:7443/tunne1
 Content-Type: text/xml
 X-Requested-With: XMLHttpRequest
 Content-Length: 83
 Cookie: finesse_ag_extension=10005; JSESSIONID=
 Connection: keep-alive
 Pragma: no-cache
 Cache-Control: no-cache

<body xmlns="http://jabber.org/protocol/httpbind

Find... (press Ctrl+Enter to highlight all)

Transformer Headers TextView SyntaxView ImageV

Raw JSON XML

```
<body xmlns="http://jabber.org/protocol/httpbind"><message  
to="47483648@fin1.uacce.local" id="/finesse/api/User/474836  
xmlns="http://jabber.org/protocol/pubsub#event"><items nod  
4752-8a1d-5adbdc74a7717"><notification xmlns="http://jab  
&lt;data&gt;  
&lt;user&gt;  
&lt;dialogs&gt;/finesse/api/User/47483648/Dialogs&lt;/dia  
&lt;extension&gt;10005&lt;/extension&gt;  
&lt;firstName&gt;isaac&lt;/firstName&gt;  
&lt;lastName&gt;Newton&lt;/lastName&gt;  
&lt;loginId&gt;47483648&lt;/loginId&gt;  
&lt;loginName&gt;isaac&lt;/loginName&gt;  
&lt;mediaType&gt;1&lt;/mediaType&gt;  
&lt;pendingState&gt;&lt;/pendingState&gt;  
&lt;roles&gt;  
&lt;role&gt;Agent&lt;/role&gt;  
&lt;/roles&gt;  
&lt;/roles&gt;  
&lt;wrapUpOnIncoming&gt;OPTIONAL&lt;/wrapUpOnInco  
&lt;/settings&gt;  
&lt;state&gt;READY&lt;/state&gt;  
&lt;stateChangeTime&gt;2019-01-11T23:56:54.783Z&lt;/  
&lt;teamId&gt;5000&lt;/teamId&gt;  
&lt;teamName&gt;Maths&lt;/teamName&gt;  
&lt;uri&gt;/finesse/api/User/47483648&lt;/uri&gt;  
&lt;/user&gt;  
&lt;/data&gt;  
&lt;event&gt;PUT&lt;/event&gt;  
&lt;requestId&gt;07f14a42-6b3c-4855-e4c9-ef50ab5e7cc6&  
&lt;source&gt;/finesse/api/User/47483648&lt;/source&gt;  
&lt;/Update&gt;</notification></item></items></event></mess
```

0:0 0/1,571 Find... (press Ctrl+Enter to hig

QuickExec] ALT+Q > type HELP to learn more

Capturing All Processes 1 / 693 https://fin1.uacce.local:7443/http-bind/

Detailliertere Ansicht der Daten:

6...	200	HTTPS	fin1.ucce.local:...	/http-bind/	1,571	text/xml...	firefo...
6...	202	HTTPS	fin1.ucce.local:...	/finesse/api/User/47...	0	no-ca...	applicatio...
6...	200	HTTPS	fin1.ucce.local:...	/desktop/theme/fine...	673	image/gif	firefo...
6...	200	HTTPS	fin1.ucce.local:...	/http-bind/	57	text/xml...	firefo...
6...	200	HTTPS	fin1.ucce.local:...	/finesse/api/SystemI...	232	no-ca...	applicatio...
6...	200	HTTPS	fin1.ucce.local:...	/http-bind/	57	text/xml...	firefo...
6...	200	HTTPS	fin1.ucce.local:...	/http-bind/	57	text/xml...	firefo...
6...	200	HTTPS	fin1.ucce.local:...	/finesse/api/SystemI...	232	no-ca...	applicatio...
6...	200	HTTPS	fin1.ucce.local:...	/http-bind/	57	text/xml...	firefo...
6...	-	HTTPS	fin1.ucce.local:...	/http-bind/	-1		firefo...
6...	200	HTTPS	fin1.ucce.local:...	/finesse/api/SystemI...	232	no-ca...	applicatio...

Antworttext für XMPP-Nachricht:

```
<body xmlns='http://jabber.org/protocol/httpbind'><message xmlns="jabber:client" from="pubsub.fin1.ucce.local"
to="47483648@fin1.ucce.local" id="/finesse/api/User/47483648__47483648@fin1.ucce.local__K7hYF"><event
xmlns="http://jabber.org/protocol/pubsub#event"><items node="/finesse/api/User/47483648"><item id="26a3e421-
4752-8a1d-5adbdc74a7717"><notification xmlns="http://jabber.org/protocol/pubsub">&lt;Update&gt;
&lt;data&gt;
&lt;user&gt;
&lt;dialogs&gt;/finesse/api/User/47483648/Dialogs&lt;/dialogs&gt;
&lt;extension&gt;10005&lt;/extension&gt;
&lt;firstName&gt;Isaac&lt;/firstName&gt;
&lt;lastName&gt;Newton&lt;/lastName&gt;
&lt;loginId&gt;47483648&lt;/loginId&gt;
&lt;loginName&gt;isaac&lt;/loginName&gt;
&lt;mediaType&gt;1&lt;/mediaType&gt;
&lt;pendingState&gt;&lt;/pendingState&gt;
&lt;roles&gt;
&lt;role&gt;Agent&lt;/role&gt;
&lt;/roles&gt;
&lt;settings&gt;
&lt;wrapUpOnIncoming&gt;OPTIONAL&lt;/wrapUpOnIncoming&gt;
&lt;/settings&gt;
&lt;state&gt;READY&lt;/state&gt;
&lt;stateChangeTime&gt;2019-01-11T23:56:54.783Z&lt;/stateChangeTime&gt;
&lt;teamId&gt;5000&lt;/teamId&gt;
&lt;teamName&gt;Maths&lt;/teamName&gt;
&lt;uri&gt;/finesse/api/User/47483648&lt;/uri&gt;
&lt;/user&gt;
&lt;/data&gt;
&lt;event&gt;PUT&lt;/event&gt;
&lt;requestId&gt;07f14a42-6b3c-4855-a4c9-af50ab5e7cc6&lt;/requestId&gt;
&lt;source&gt;/finesse/api/User/47483648&lt;/source&gt;
&lt;/Update&gt;</notification></item></items></event></message></body>
```

Wireshark verwenden

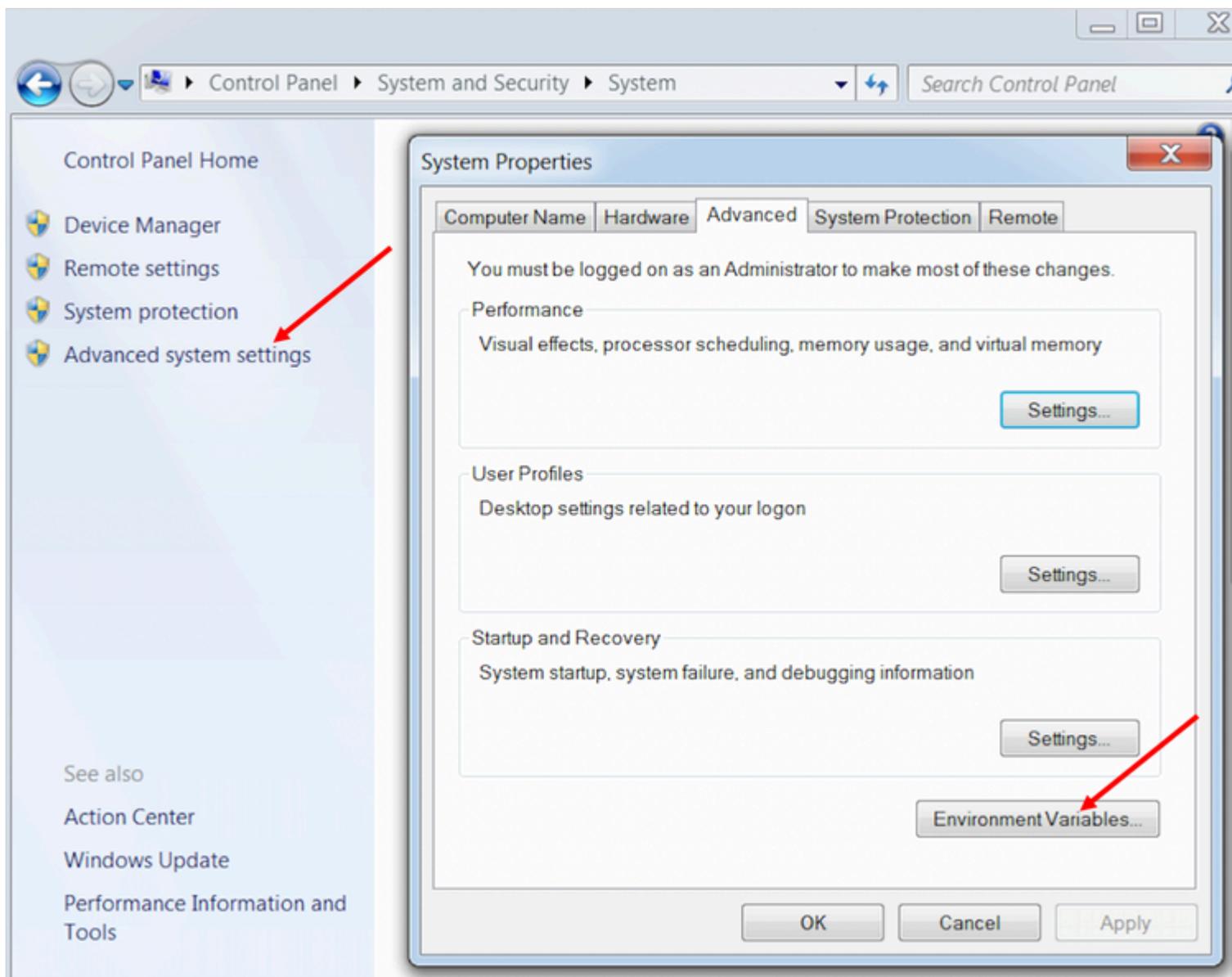
Wireshark ist ein häufig verwendetes Paket-Sniffing-Tool, mit dem HTTPS-Datenverkehr gesniffen und decodiert werden kann. HTTPS-Datenverkehr ist HTTP-Datenverkehr, der über TLS (Transport Layer Security) gesichert wird. TLS gewährleistet Integrität, Authentifizierung und Vertraulichkeit zwischen zwei

Hosts. Es wird häufig in Webanwendungen verwendet, kann jedoch mit jedem Protokoll verwendet werden, das TCP als Transportschichtprotokoll verwendet. Secure Sockets Layer (SSL) ist die frühere Version des TLS-Protokolls, das nicht mehr verwendet wird, da es unsicher ist. Diese Namen werden häufig synonym verwendet, und der Wireshark-Filter, der für SSL- oder TLS-Datenverkehr verwendet wird, ist ssl.

Achtung: Die angegebene Beispielformatierung gilt für Wireshark 2.6.6 (v2.6.6-0-gdf942cd8) und Mozilla Firefox 64.0.2 (32-Bit) unter Windows7 x64 in einer Laborumgebung. Diese Verfahren können nicht für alle Versionen von Fiddler, alle Browser oder alle Computer-Betriebssysteme verallgemeinert werden. Wenn Ihr Netzwerk in Betrieb ist, stellen Sie sicher, dass Sie die potenziellen Auswirkungen jeder Konfiguration kennen. Weitere Informationen finden Sie in der [offiziellen Wireshark SSL-Dokumentation](#). Wireshark 1.6 oder höher ist erforderlich.

Hinweis: Diese Methode kann nur für Firefox und Chrome funktionieren. Diese Methode funktioniert nicht für Microsoft Edge.

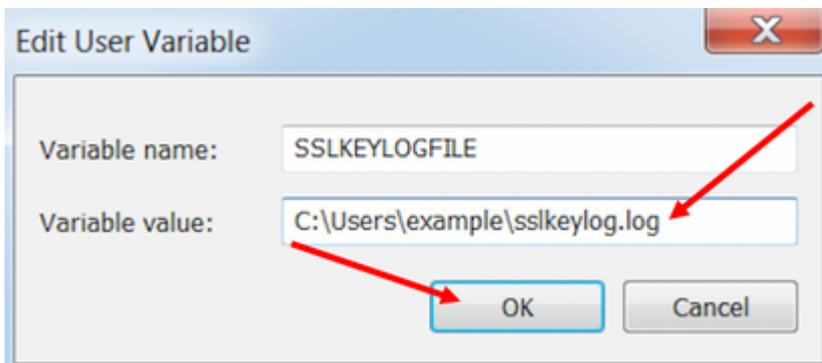
Schritt 1: Navigieren Sie auf dem Windows-PC des Agenten zu **Systemsteuerung > System und Sicherheit > System > Erweiterte Systemeinstellungen Umgebungsvariablen...**



Schritt 2: Navigieren Sie zu **Benutzervariablen für Benutzer <Benutzername> > Neu...**

Erstellen Sie eine Variable mit dem Namen **SSLKEYLOGFILE**.

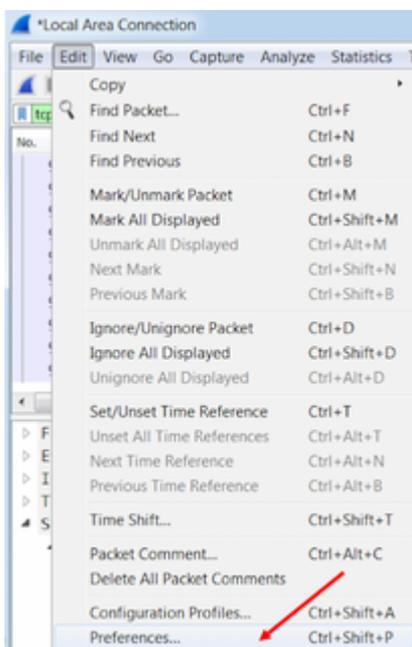
Erstellen Sie eine Datei, um den SSL-Premaster-Schlüssel in einem privaten Verzeichnis zu speichern:
SSLKEYLOGFILE=</path/to/private/directory/with/logfile>



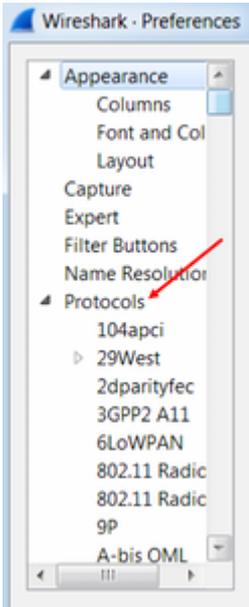
Hinweis: Erstellen Sie eine Systemvariable anstelle einer Benutzervariablen und/oder speichern Sie die Datei in einem nicht privaten Verzeichnis, aber dann können alle Benutzer im System auf den premaster secret zugreifen, was weniger sicher ist.

Schritt 3: Wenn Firefox oder Chrome geöffnet sind, schließen Sie die Anwendungen. Nach dem erneuten Öffnen können sie mit dem Schreiben in die SSLKEYLOGFILE beginnen.

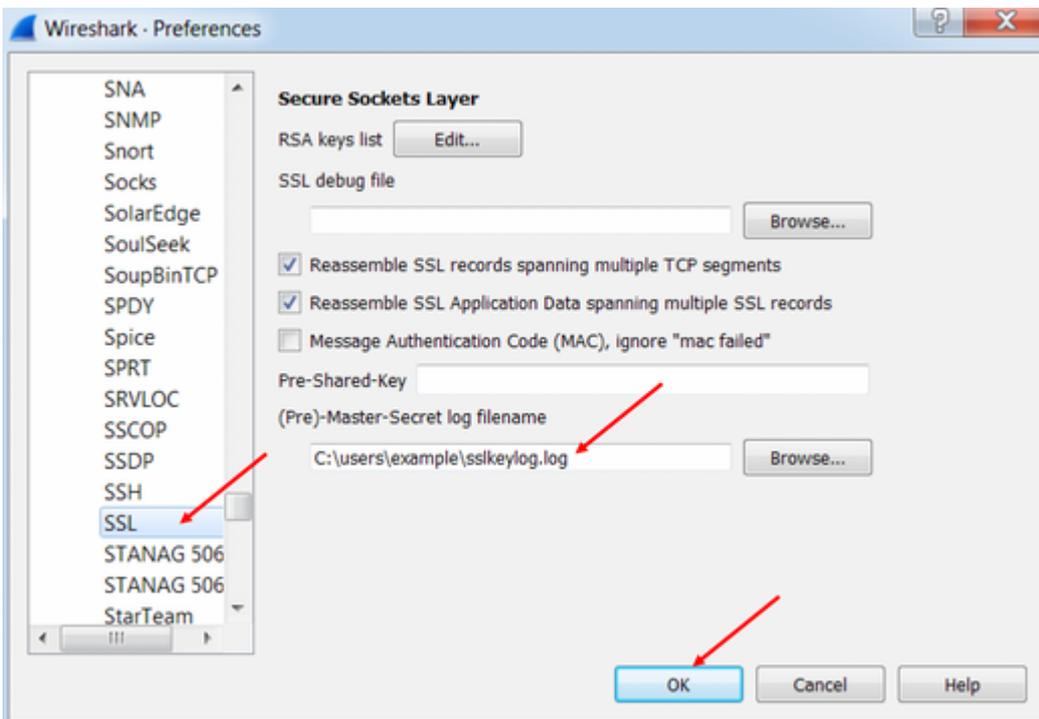
Schritt 4: Navigieren Sie in Wireshark zu **Bearbeiten > Voreinstellungen...**



Navigieren Sie zu **Protokolle > SSL**.



Schritt 5: Geben Sie den in Schritt 2 konfigurierten Speicherort der Datei mit dem geheimen Premaster-Protokoll ein.



Schritt 6: Verwenden Sie Wireshark filter **tcp.port==7443 && ssl**, die sichere HTTP-Kommunikation zwischen dem Finesse-Client und Finesse-Server (Notification Service) wird entschlüsselt gesehen.

```
Transmission Control Protocol, Src Port: 54979, Dst Port: 7443 Seq: 21265, Ack: 42841, Len: 565
Secure Sockets Layer
  TLSv1.2 Record Layer: Application Data Protocol: Application Data
    Content Type: Application Data (23)
    Version: TLS 1.2 (0x0303)
    Length: 560
    Encrypted Application Data: 1e001ee88fc1c9a026b0385007608afdfb46c0d4a277faa8...

0010  20 48 54 54 50 2f 31 2e 31 0d 0a 48 6f 73 74 3a  HTTP/1.1 -Host:
0020  20 66 69 6e 31 2e 75 63 63 65 2e 6c 6f 63 61 6c  fin1.uce.local
0030  3a 37 34 34 33 0d 0a 55 73 65 72 2d 41 67 65 6e  :7443 -User-Agen
0040  74 3a 20 4d 6f 7a 69 6c 6c 61 2f 35 2e 30 20 28  t: Mozilla/5.0 (
0050  57 69 6e 64 6f 77 73 20 4e 54 20 36 2e 31 3b 20  Windows NT 6.1;
0060  57 4f 57 36 34 3b 20 72 76 3a 36 34 2e 30 29 20  WOW64; rv:64.0)
0070  47 65 63 6b 6f 2f 32 30 31 30 30 31 30 31 20 46  Gecko/2010101 Firefox/6
0080  69 72 65 66 6f 78 2f 36 34 2e 30 0d 0a 41 63 63  irefox/6.4.0 -Acc
0090  65 70 74 3a 20 74 65 78 74 2f 70 6c 61 69 6e 2c  ept: text/plain,
00a0  20 2a 2f 2a 3b 20 71 3d 30 2e 30 31 0d 0a 41 63  /*; q=0.01 -Ac
00b0  63 65 70 74 2d 4c 61 6e 67 75 61 67 65 3a 20 65  cept-Language: e
00c0  6e 2d 55 53 2c 65 6e 3b 71 3d 30 2e 35 0d 0a 41  n-US,en;q=0.5 -A

Frame (619 bytes) Decrypted SSL (513 bytes)
wireshark_E6642FDE-A01F-4115-B2E4-85157AB917CB_20190125155406_a06084.pcapng Packets: 127485 · Display
```

Verwandte Fehler

- Cisco Bug-ID [CSCva7280](#) - Finesse Tomcat und Openfire Crash für ungültige XML-Zeichen
- Cisco Bug-ID [CSCva7235](#) - UCCX: Finesse Tomcat und Openfire-Absturz wegen ungültiger XML-Zeichen

Zugehörige Informationen

- [XMPP-Spezifikationen](#)
- [XEP-0124: BOSH](#)
- [XEP-0060: Veröffentlichen-Abonnieren](#)
- [Firefox-Webkonsole](#)
- [Microsoft Edge-Webkonsole](#)
- [Chrome-Webkonsole](#)
- [Windows PowerShell](#)
- [Windows-Leistungsüberwachung](#)
- [Fehlerbehebung bei Unterbrechungen von Eingabe- und Ausgabewarteschlangen](#)
- [Windows Task-Manager](#)
- [Mac-Terminal](#)
- [MAC-Aktivitätsüberwachung](#)
- [Fiddler-Download](#)
- [Fiddler-Konfiguration](#)
- [Wireshark-Download](#)
- [Wireshark SSL-Verschlüsselung](#)
- [Technischer Support und Dokumentation für Cisco Systeme](#)

Informationen zu dieser Übersetzung

Cisco hat dieses Dokument maschinell übersetzen und von einem menschlichen Übersetzer editieren und korrigieren lassen, um unseren Benutzern auf der ganzen Welt Support-Inhalte in ihrer eigenen Sprache zu bieten. Bitte beachten Sie, dass selbst die beste maschinelle Übersetzung nicht so genau ist wie eine von einem professionellen Übersetzer angefertigte. Cisco Systems, Inc. übernimmt keine Haftung für die Richtigkeit dieser Übersetzungen und empfiehlt, immer das englische Originaldokument (siehe bereitgestellter Link) heranzuziehen.