Traço do início de uma sessão do agente da fineza com o uso dos logs

Índice

Introdução

Este original descreve o processo envolvido em um início de uma sessão do agente através do sistema da fineza com os arquivos de registro. É importante compreender o fluxo de mensagem entre os componentes diferentes da fineza, o server da integração de telefonia e computador (CTI), e o desktop de cliente de modo que você possa com sucesso pesquisar defeitos edições.

Pré-requisitos

Requisitos

Cisco recomenda que você tem o conhecimento da fineza de Cisco e da alerta de comando CLI do sistema operacional da Voz (VOS).

Componentes Utilizados

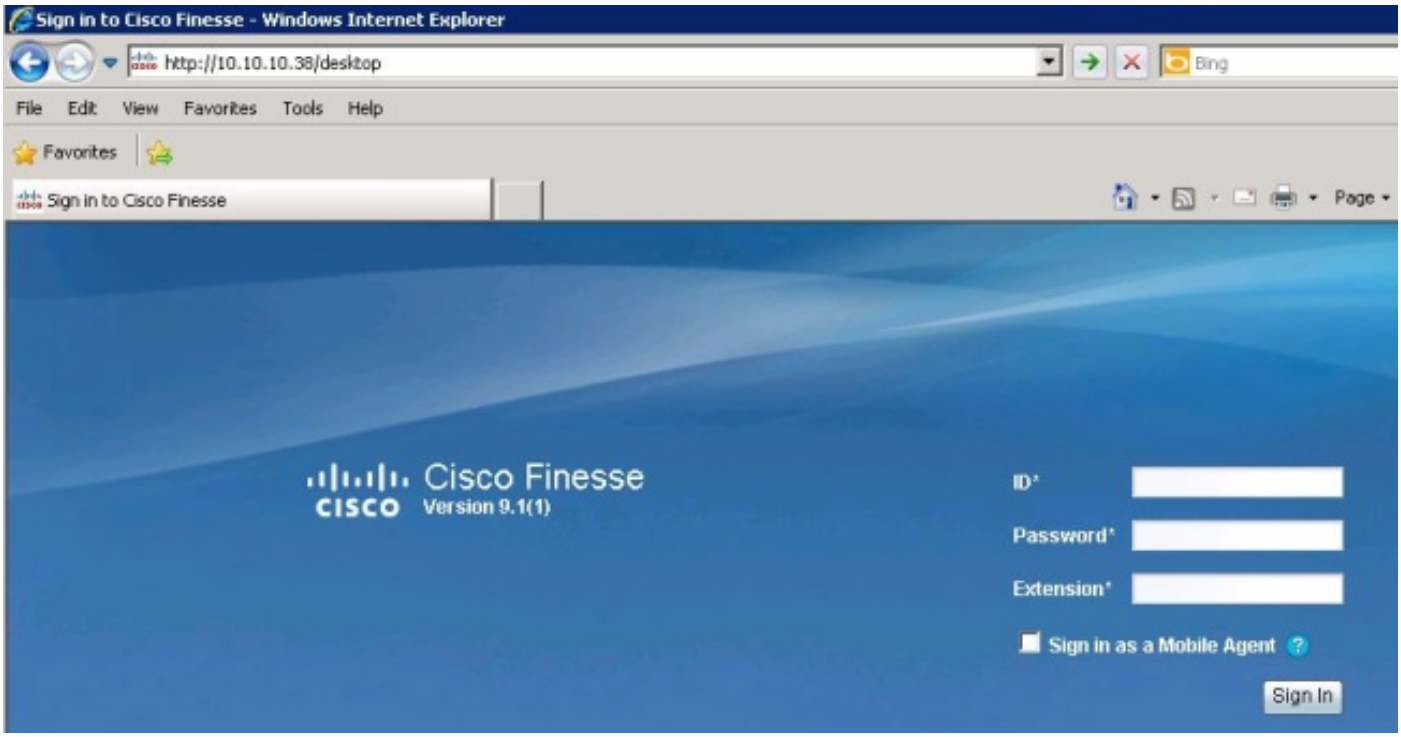
A informação neste documento é baseada na versão 9.1(1) da fineza de Cisco.

As informações neste documento foram criadas a partir de dispositivos em um ambiente de laboratório específico. Todos os dispositivos utilizados neste documento foram iniciados com uma configuração (padrão) inicial. Se a sua rede estiver ativa, certifique-se de que entende o impacto

potencial de qualquer comando.

Lance a área de trabalho do agente

A fim lançar a área de trabalho do agente, copie esta URL no navegador da Web: **fineza server>/desktop do <your de http://**. Na versão 9.1 da fineza, o HTTP ou o HTTPS são apoiados.



A fineza usa Tomcat como o servidor de Web. Quando você lança seu navegador da Web, o pedido está feito à fineza para apresentar-lhe a área de trabalho do agente. O comando do **localhose_access_log de** Cisco Tomcat mostra o pedido carregar a área de trabalho do agente.

```
10.10.10.211 10.10.10.211 - - 80 GET /  HTTP/1.1 302 - 141
10.10.10.211 10.10.10.211 - - 80 GET /desktop/container/ HTTP/1.1 200 4541 185
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/css/base.css
HTTP/1.1 200 3093 7
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/css/login.css
HTTP/1.1 200 2185 0
10.10.10.211 10.10.10.211 - - 80 GET /desktop/js/Logon.js HTTP/1.1 200 1745 1
10.10.10.211 10.10.10.211 - - 80 GET /desktop/js/utilities/Cookies.js HTTP/1.1
200 2390 0
10.10.10.211 10.10.10.211 - - 80 GET /desktop/thirdparty/jquery/js/jquery.tools.
min.js HTTP/1.1 200 15699 1
10.10.10.211 10.10.10.211 - - 80 GET /desktop/thirdparty/jquery/js/jquery-1.5.
min.js HTTP/1.1 200 84523 7
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/images/sprite_
buttons.png HTTP/1.1 200 3297 0
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/images/help.png
HTTP/1.1 200 830 0
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/images/cisco_logo.
png HTTP/1.1 200 760 0 200 2205 1
10.10.10.211 10.10.10.211 - - 80 GET /desktop/theme/finesse/images/bg.jpg
HTTP/ 1.1 200 32222 4
```

Credenciais do início de uma sessão do agente

Agora que a área de trabalho do agente foi apresentada, você incorpora suas credenciais do início de uma sessão. Antes que a fineza possa enviar a solicitação de login ao CTI Server, o cliente precisa de estabelecer Bidirecional-córregos sobre a conexão síncrono HTTP (PARVOÍCE). A fim estabelecer a conexão do PARVOÍCE, o cliente pede primeiramente a informação de sistema do server da fineza.

SystemInfo

O desktop do cliente fez um estado representacional transferir o pedido da interface de programação de aplicativo (do RESTO) (API) a esta URL: **/finesse/api/SystemInfo**. Tome a nota do **nocache=**. Este ID exclusivo é usado a fim seguir este pedido através do sistema. **Retornado com status=200** indica que o pedido esteve recebido com sucesso.

```
Container : [ClientServices] SystemInfo: requestId='undefined', Making REST
request: method=GET, url='/finesse/api/SystemInfo?nocache=1366756802163'18:40:03:
Container : [ClientServices] SystemInfo: requestId='undefined', Returned
with status=200
```

Se você não tem clientlogs mas você precisa de seguir o pedido, você pode procurar o **localhost_access_log de** Tomcat a fim determinar quando o pedido do RESTO API foi feito e encontrar o identificador exclusivo.

```
127.0.0.1 127.0.0.1 - - 80 GET /finesse/api/SystemInfo ?nocache=1366756802163
HTTP/1.1 200 336 120 10.10.10.211 10.10.10.211 2001 - 80 GET /gadgets/makeRequest
?refresh=3600&url=http%3A%2F%2Flocalhost%2Ffinesse%2Fapi%2FSystemInfo%3Fnocache%
3D1366756802163&httpMethod=GET&headers=Authorization%3DBasic%2520MjAwMToyMDAx%
26locale%3Den_US&postData=&authz=&st=&contentType=TEXT&numEntries=3&getSummaries
=false&signOwner=true&signViewer=true&gadget=undefined&container=default&
bypassSpecCache=&getFullHeaders=false HTTP/1.1 200 659 596
```

API_REQUEST

Tomcat envia este pedido API ao repositório do aplicativo de web do RESTO API da fineza (GUERRA). A fim encontrar os logs do RESTO API da fineza, procure os webservices da fineza registram pelo timestamp ou pelo nocache ID a fim encontrar o API_REQUEST. Este log mostra o **REQUEST_START**, o **REQUEST_URL**, o **REQUEST_END**, e o **elapsed_time que o** sistema tomou para terminar o pedido.

```
%CCBU_http-8080-7-6-REQUEST_START: %[method_name=GET][parameter_name=
{ nocache=[1366756802163],  }][resource_name=/SystemInfo][usr=]: Request start
%CCBU_http-8080-7-6-API_REQUEST: %[REQUEST_URL=SystemInfo][agent_id=][request_
identifier=][request_method=systemInfo.GET][request_parameters=]: Request from
client to webservice api
%CCBU_http-8080-7-6-REGISTER_API_STATS_OBJECT: %[resource_name=com.cisco.ccbu:
category=WebAppStats,component0=SystemInfo-GET]: Registered new api stats object
for new request type. %CCBU_http-8080-7-6-REQUEST_END: %[elapsed_time=98]: Request complete
```

O índice retornado ao cliente pelo pedido do RESTO API recuperar a informação de sistema é mostrado aqui. Este é encontrado nos logs do cliente (agente).

```
%CCBU_http-8080-7-6-REQUEST_START: %[method_name=GET][parameter_name=
{ nocache=[1366756802163],  }][resource_name=/SystemInfo][usr=]: Request start
%CCBU_http-8080-7-6-API_REQUEST: %[REQUEST_URL=SystemInfo][agent_id=][request_
identifier=][request_method=systemInfo.GET][request_parameters=]: Request from
client to webservice api
%CCBU_http-8080-7-6-REGISTER_API_STATS_OBJECT: %[resource_name=com.cisco.ccbu:
category=WebAppStats,component0=SystemInfo-GET]: Registered new api stats object
for new request type. %CCBU_http-8080-7-6-REQUEST_END: %[elapsed_time=98]: Request complete
```

Estabeleça a conexão do PARVOÍCE

O SystemInfo mostra os server preliminares e secundários da fineza, o estado da fineza como o **IN_SERVICE**, o **xmppDomain**, e o **xmppPubSubDomain**. O cliente tem agora bastante informação a fim estabelecer uma conexão do PARVOÍCE.

```
18:40:03: Container : PageServices.init().onLoad: System info status: IN_SERVICE
18:40:03: Container : PageServices.init(): Establishing BOSH connection...
18:40:03: Container : PageServices.init(): Starting timeout and poller...
18:40:04: Container : [ClientServices] MasterPublisher._connInfoHandler()
- Connection status: connecting
18:40:04: Container : [ClientServices] MasterPublisher._connInfoHandler()
- Connection status: disconnected
18:40:04: Container : PageServices._onDisconnect(): retryCount=0, retrying...
18:40:04: Container : [ClientServices] MasterPublisher._connInfoHandler()
- Connection status: connecting
18:40:05: Container : [ClientServices] MasterPublisher._connInfoHandler()
- Connection status: connected
18:40:05: Container : PageServices.onLoad(): BOSH established!
```

O cliente é subscrito com sucesso ao objeto da fineza (nó) **/finesse/api/User/2001** uma vez que a conexão do PARVOÍCE é estabelecida.

Quando a conexão do PARVOÍCE do cliente é estabelecida, o log dos webservices recebe uma mensagem **PRESENCE_NOTIFICATION do** cliente. Este **PRESENCE_TYPE** indica somente que o cliente está disponível para receber **eventos XMPP** e não não tem nada fazer com a disponibilidade de agente na empresa unificada do centro de contato (UCCE). Recorde que o agente não está assinado dentro ainda.

> **Note**: Você vê somente as mensagens **PRESENCE_TYPE** quando um cliente estabelece uma conexão do PARVOÍCE ou quando a conexão do PARVOÍCE de um cliente está desligada. Quando a conexão do PARVOÍCE do cliente for desligada, os indicadores **PRESENCE_TYPE** como não disponíveis.

Está aqui o evento de notificação no log dos websservices:

```
%CCBU_Smack Listener Processor (1)-6-PRESENCE_NOTIFICATION_RECIEVED:
%[FROM JID=2001@uccefinesse138.vmload.cvp/desktop]
[PRESENCE_TYPE=available]: Finesse received a presence notifcation
```

Login do agente

Agora que o cliente estabeleceu a conexão do PARVOÍCE, o processo do login começa. O cliente

faz um outro pedido do RESTO API a fim obter a informação sobre o usuário atual. A fim fazer este pedido, navegue a esta URL: **/finesse/api/User/2001** e incorpora o method=GET.

Porque este é um pedido diferente API, o **nocache ID** é diferente. Assim, a fim seguir este pedido, você precisa de usar este ID novo.

```
Container : PageServices.onLoad(): BOSH established! Commencing sign-in process
Container : [ClientServices] User: requestId='undefined', Making REST request:
method=GET, url='/finesse/api/User/2001?nocache=1366756805180
'18:40:05: Container : [ClientServices] User: requestId='undefined',
Returned with status=200,
```

Você pode encontrar este pedido no **localhost_access_log de** Tomcat se necessário. É aqui como você o encontra no log dos webservices:

```
%CCBU_http-8080-7-6-REQUEST_START: %[method_name=GET][parameter_name={ nocache=
[1366756805180],  }][resource_name=/User/2001][usr=2001]: Request start
%CCBU_http-8080-7-6-API_REQUEST: %[REQUEST_URL=User/2001][agent_id=2001]
[request_identifier=null][request_method=user.GET][request_parameters=2001]:
Request from client to webservice api
```

Está aqui o pedido no log dos Serviços de notificação. Tome a nota da **aprovação HTTP/1.1 200**.

> **Note**: O log da notificação de Cisco é apenas para fins informativos. Se você permite a notificação da fineza de Cisco que registra, impacta o desempenho.

```
>> "GET /finesse/api/User/2001 HTTP/1.1[\r][\n]"
Adding Host request header
>>"Authorization: Basic MjAwMToyMDAx[\r][\n]"
>>"User-Agent: Jakarta Commons-HttpClient/3.1[\r][\n]"
>>"Host: localhost:8080[\r][\n]"
>>"[\r][\n]"
<<"HTTP/1.1 200 OK[\r][\n]"
<<"HTTP/1.1 200 OK[\r][\n]"
<<"Pragma: No-cache[\r][\n]"
<<"Cache-Control: no-cache[\r][\n]"
```

Agora que o serviço de notificação tem o pedido, afixa a informação para este usuário. Está aqui o CARGO do log do serviço de notificação que vai ao cliente:

```
>> "GET /finesse/api/User/2001 HTTP/1.1[\r][\n]"
Adding Host request header
>>"Authorization: Basic MjAwMToyMDAx[\r][\n]"
>>"User-Agent: Jakarta Commons-HttpClient/3.1[\r][\n]"
>>"Host: localhost:8080[\r][\n]"
>>"[\r][\n]"
<<"HTTP/1.1 200 OK[\r][\n]"
<<"HTTP/1.1 200 OK[\r][\n]"
<<"Pragma: No-cache[\r][\n]"
<<"Cache-Control: no-cache[\r][\n]"
```

Este **evento XMPP**, que é o **agente 2001 n**este exemplo, é enviado a todos os clientes da assinatura. O Javascript no cliente recebe o **evento XMPP**, e o evento é enviado ao dispositivo dentro do cliente. Estão aqui os clientlogs que mostram o índice da resposta:

```
Commencing sign-in process18:40:05: Container : [ClientServices] User: requestId=
```

```
'undefined', Maurl='/finesse/api/User/2001?nocache=1366756805180'18:40:05:
Container : [ClientServices] User: requestId='undefined', Returned with status=200,
content='<User> king REST request: method=GET,
<dialogs>/finesse/api/User/2001/Dialogs</dialogs>
<extension></extension>
<firstName>Mickey</firstName>
<lastName>Mouse</lastName>
<loginId>2001</loginId>
<loginName>mmouse</loginName>
<roles>
<role>Agent</role>
</roles>
<state>LOGOUT</state>
<stateChangeTime></stateChangeTime>
<teamId>5000</teamId>
<teamName>Minnies_Team</teamName>
<uri>/finesse/api/User/2001</uri>
</User>
```

Execute o início de uma sessão

Agora o cliente está pronto para executar o início de uma sessão. Observe o **RequestID**. O RequestID é enviado no corpo do pedido. Você usa este RequestID a fim seguir a solicitação de login ao **RESTO API > CTI > RESTO API > serviço de notificação > resposta** de volta ao cliente. Este pedido é POSTA, assim que significa que o cliente está pedindo uma ATUALIZAÇÃO ou uma mudança a seu estado atual.

```
Container : SignIn.handleUserLoad(): Performing login: extn=2003 18:40:05:
Container : [ClientServices] User: requestId='6e210ca9-5786-43bc-babf-
64a397a6057f',
</data>
<event>PUT</event>
<requestId>6e210ca9-5786-43bc-babf-64a397a6057f</requestId>
<source>/finesse/api/User/2001</source>
</Update>
```

O RESTO API da fineza recebe este pedido do cliente. Então, o API envia um **SetAgentStateReq ao** CTI Server.

```
%CCBU_http-8080-7-6-API_REQUEST: %[REQUEST_URL=User/2001][agent_id=2001]
[request_identifier=6e210ca9-5786-43bc-babf-64a397a6057f][request_method=
user.PUT][request_parameters= extension:2003 state:LOGIN]: Request from
client to webservice api
%CCBU_http-8080-7-6-REGISTER_API_STATS_OBJECT: %[resource_name=com.cisco.ccbu:
category=WebAppStats,component0=User-[id]-PUT]: Registered new api stats object
for new request type.
%CCBU_http-8080-7-6-REQUEST_END: %[elapsed_time=8]: Request complete
%CCBU_pool-5-thread-4-6-MESSAGE_TO_CTI_SERVER: %[cti_message=Invoke id :20 ,
agentstate : 0, workmode : 0, reason code: -15532, forceflag :1, agentcapacity:
0, agentext: 2003, agentid: 2001][cti_message_name=SetAgentStateReq]:
Message going to the backend cti server
```

O CTI Server recebe o pedido.

```
Trace: AGENT_EVENT: ID=2001 Periph=5001 Ext=2003 Inst=2003 Sig=
Trace: SkgState=LOGIN SkgDuration=0 OverallState=NOT_READY OverallDuration=0
Reason=0
Trace: MRDID=1 NumTasks=0 MaxTaskLimit=1 AgtMode=1 AGTAvailabilityStatus=0
```

```
ICMAgtID=5001
Trace: SkTgtID=5001 SkGrpNo=0x0 SkGrpID=5006 NumLines=0 CurLine=0 ClientStatus=
0x0 Direction=0
```

Uma vez que o agente é entrado com um estado de **NOT_READY**, o CTI Server envia **AGENT_STATE-EVENT à** fineza.

```
MsgType:AGENT_STATE_EVENT (MonitorID:0 PeripheralID:5001 SessionID:0x0
PeripheralType:EnterpriseAgent SkillGroupState:LOGIN StateDuration:0
SkillGroupNumber:85881 SkillGroupID:5000 SkillGroupPriority:0 AgentState:
NOT_READY EventReasonCode:0 MRDID:1 NumTasks:0 AgentMode:1 MaxTaskLimit:1
ICMAgentID:5001 AgentAvailabilityStatus:0 NumFltSkillGroups:0 Direction:0
ClientSignature:""AgentID:"2001" AgentExtension:"2003" AgentInstrument:"2003"
RemaskNumMasks:1 RemaskInstrument:"2003" RemaskExtension:"2003" RemaskCallId:
0xffffffff RemaskFunctionFlag:<0x38> <LogoutCodeReq,NotRdyCodeReq,WrapDataReq>
RemaskCallMask:<0x21000000> <MC,Emerg> RemaskAgentMask:<0x0a000000> <
Logout,Avail> )Trace: AGENT_EVENT: ID=2001 Periph=5001 Ext=2003 Inst=2003 Sig=
Trace: SkgState=LOGIN SkgDuration=0 OverallState=NOT_READY OverallDuration=0
Reason=0 Trace: MRDID=1 NumTasks=0 MaxTaskLimit=1 AgtMode=1
AGTAvailabilityStatus=0 ICMAgtID=5001
```

Está aqui o log dos webservices que recebeu o evento do CTI Server. Recorde que você vê o **mensagem bruta do** CTI Server primeiramente, e então você veja a mensagem **descodificada**.

```
%CCBU_CTIMessageEventExecutor-0-6-DECODED_MESSAGE_FROM_CTI_SERVER: %[cti_message
=CTIAgentStateEvent [skillGroupState=0 (LOGIN), stateDuration=0, skillGroupNumber
=85881, skillGroupPriority=0, agentState=2 (NOT_READY), eventReasonCode=0,
numFltSkillGroups=0,CTIClientSignature=, agentID=2001, agentExtension=2003,
agentInstrument=2003, agentID_ Long=null, duration=null, nextAgentState=null,
fltSkillGroupNumberList=[], fltSkill GroupIDList=[], fltSkillGroupPriorityList=[],
fltSkillGroupStateList=[]]CTIMessageBean [invokeID=null, msgID=30, timeTracker=
{"id":"AgentStateEvent","CTI_MSG_RECEIVED":1366756808374,
"CTI_MSG_DISPATCH":1366756808375}, msgName=AgentStateEvent, deploymentType=CCE]]
[cti_response_time=1]: Decoded Message to Finesse from backend cti server
```

Agora que a fineza recebeu o **AgentStateEvent do** CTI Server, o evento precisa **de ser publicado ao** serviço de notificação de modo que o cliente receba a ATUALIZAÇÃO. A única maneira para que o agente saiba que a sua/seu estado mudou é recebendo este **evento XMPP**. A fineza converte o **AgentStateEvent a XMPP** e envia o **XMPP ao** serviço de notificação. Observe que o evento é **POSTA**, e o RequestID está no payload.

```
%CCBU_pool-5-thread-5-6-XMPP_PUBLISH_ASYNCHRONOUS: %[NodeId=/finesse/api/User/
2001][Payload=<Update><data><user><dialogs>/finesse/api/User/2001/Dialogs
</dialogs><extension>2003</extension><firstName>Mickey</firstName><lastName>
Mouse</lastName><loginId>2001</loginId><loginName>mmouse</loginName>
<reasonCodeId>-1</reasonCodeId><roles><role>Agent</role></roles><state>NOT_READY
</state><stateChangeTime>2013-04-23T22:40:08Z</stateChangeTime><teamId>5000
</teamId><teamName>Minnies_Team</teamName><uri>/finesse/api/User/2001
</uri></user></data><event>PUT</event><requestId>6e210ca9-5786-43bc-babf-
64a397a6057f </requestId><source>/finesse/api/User/2001</source></Update>]:
Publishing XMPP Message Asynchronously
```

Aqui, o serviço de notificação recebe a ATUALIZAÇÃO. Mesmo que a mensagem diga **falhado ao pacote de rota a JID**, uma mensagem que um evento esteve publicado é enviada ao usuário.

```
RoutingTableImpl: Failed to route packet to JID: 2001@uccefinesse138.vmload.cvp/
User packet: <message from="pubsub.uccefinesse138.vmload.cvp" to=
"2001@uccefinesse138.vmload.cvp/ User" id="/finesse/api/User/
2001__2001@uccefinesse138.vmload.cvp__VI1B2"><event xmlns=
```

"http://jabber.org/protocol/pubsub#event"><items node="/finesse/api/User/2001">
<item id="1su0Keff8M2irdS"><notification xmlns="http://jabber.org/protocol/pubsub">
<Update>

Está aqui o corpo da mensagem:

<data>
<user>
<dialogs>/finesse/api/User/2001/Dialogs</dialogs>
<extension>2003</extension>
<firstName>Mickey</firstName>
<lastName>Mouse</lastName>
<loginId>2001</loginId>
<loginName>mmouse</loginName>
<reasonCodeId>-1</reasonCodeId>
<roles>
<role>Agent</role>
</roles>
<state>NOT_READY</state>
<stateChangeTime>2013-04-23T22:40:08Z</stateChangeTime>
<teamId>5000</teamId>
<teamName>Minnies_Team</teamName>
<uri>/finesse/api/User/2001</uri>
</user>
</data>
<event>**PUT**</event>
<requestId>**6e210ca9-5786-43bc-babf-64a397a6057f&lt**;/requestId>
<source>/finesse/api/User/2001</source>
</Update></notification></item></items></event></message>

Como antes, a mensagem XMPP é recebida pelo cliente e entregada ao dispositivo do cliente. Observe que o cliente recebe o evento com o RequestID original na mensagem.

Returned with status=202, content=''18:40:05: Container : [ClientServices]
MasterPublisher._eventHandler() - Received event on node '/finesse/api/User/
2001': <Update>
<data>
<user>
<dialogs>/finesse/api/User/2001/Dialogs</dialogs>
<extension>2003</extension>
<firstName>Mickey</firstName>
<lastName>Mouse</lastName>
<loginId>2001</loginId>
<loginName>mmouse</loginName>
<reasonCodeId>-1</reasonCodeId>
<roles>
<role>Agent</role>
</roles>
<state>**NOT_READY**</state>
<stateChangeTime>2013-04-23T22:40:08Z</stateChangeTime>
<teamId>5000</teamId>
<teamName>Minnies_Team</teamName>
<uri>/finesse/api/User/2001</uri>
</user>
</data>
<event>**PUT**</event>
<requestId>**6e210ca9-5786-43bc-babf-64a397a6057f**</requestId>
<source>/finesse/api/User/2001</source>
</Update>

O cliente é entrado agora com sucesso.

```
Container : SignIn._triggerLoggedIn(): Successfully logged in!18:40:05
```

Códigos da saída, códigos de motivo, agenda de telefones

Agora o cliente precisa de recuperar dados agente-específicos, tais como códigos da saída, códigos de motivo, e agenda de telefones. Está aqui o pedido para essa informação feita ao cliente.

```
Container : SignIn._triggerLoggedIn(): Successfully logged in!18:40:05
```

A mesma lógica aplica a estes pedidos. Mantenha na mente que os códigos de motivo e a agenda de telefones da fineza são armazenados no base de dados da fineza, não em UCCE.