Troubleshoot Call Setup Issues with VTR and VTRI

Contents

Introduction

This document describes how to use Verify Translation and Routing (VTR) tool in the Application Server (AS) to troubleshoot common call setup issues.

.

Background Information

BroadWorks Application Server has two embedded tools that can be used to troubleshoot call setup issues: VTR and Verify Translation and Routing INVITE (VTRI). The purpose of these two tools is the same, the difference is on the commands input:

- VTR takes the caller details (such as userId or phone number), as well as destination number as input,

- VTRI takes the SIP INVITE message as input.

Caution: VTR and VTRI commands are also available in the Network Server (NS), but the output structure is different from the AS. This document focuses on AS VTR(I) and cannot be used as a reference during work with NS VTR(I).

Full command reference for AS VTR is available in Application Server VTR Administration Guide.

Prerequisites

Requirements

Cisco recommends that you have knowledge of these:

- Session Initiation Protocol (SIP) signalling.
- Call processing mechanism of the Application Server, along with the available services and their configuration.

Components Used

The information in this document is based on BroadWorks AS version R24. However, behavior for other software versions is similar.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Usage of VTR(I) Tool

VTR analysis can be triggered both from Command Line Interface (CLI) and from the Web interface.

Run VTR(I) Tool from CLI

In order to run VTR from Web GUI, use these steps:

Step 1. Log into BroadWorks CLI with your bwadmin credentials.

Step 2. Start BroadWorks CLI and navigate to **AS_CLI/ASDiagnostic/Diag:**

```
AS_CLI> cd ASDiagnostic/Diag
```

Step 3. In order to use VTRI, run the **vtri** command, followed by SIP INVITE, and the period character in new line. For example:

```
AS_CLI/ASDiagnostic/Diag> vtri Enter a SIP message. When complete, enter a single period (.) on a line to start verifying the translation. INVITE sip:2012@mleus.lab SIP/2.0 Via: SIP/2.0/UDP 10.61.205.219:58300;rport;branch=z9hG4bKPjgINPvPUvoBT57iTOBPsgCfEqE5GX1aj7 Max-Forwards: 70 From: "Marek Leus" <sip:5403362011@mleus.lab>;tag=6fU.VlLrWc6WI3JU8jWKS.25yeoWEhpc To: sip:2012@mleus.lab Contact: "Marek Leus" <sip:5403362011@10.61.205.219:58300;ob> Call-ID: dTUVBWON9UjmftpGCOoJzhLfbajBm11C CSeq: 6492 INVITE Route: <sip:10.48.93.126;lr> Allow: PRACK, INVITE, ACK, BYE, CANCEL, UPDATE, INFO, SUBSCRIBE, NOTIFY, REFER, MESSAGE, OPTIONS Supported: replaces, 100rel, norefersub User-Agent: Telephone 1.6 Content-Type: application/sdp Content-Length: 480 v=0 o=- 3883737105 3883737105 IN IP4 10.61.205.219 s=pjmedia b=AS:117 t=0 0 a=X-nat:0 m=audio 4012 RTP/AVP 96 9 8 0 101 102 c=IN IP4 10.61.205.219 b=TIAS:96000 a=rtcp:4013 IN IP4 10.61.205.219 a=sendrecv a=rtpmap:96 opus/48000/2 a=fmtp:96 useinbandfec=1 a=rtpmap:9 G722/8000 a=rtpmap:8 PCMA/8000 a=rtpmap:0 PCMU/8000 a=rtpmap:101 telephone-event/48000 a=fmtp:101 0-16 a=rtpmap:102 telephone-event/8000 a=fmtp:102 0-16 a=ssrc:2039250127 cname:43ec7f3b5b951d53 .
```

Step 4. In order to use VTR, run the **vtr** command, followed by details of caller and destination. For example:

```
# Command reference: AS_CLI/ASDiagnostic/Diag> h vtr This command is used to perform a test call and see the routing results. Please be careful when performing test calls, it may have an impact on existing real calls or cause user migration. Parameters description: origType : The type of origination used to trigger the VTR request. linePort : The lineport of the originating user. bwphone : The originating BroadWorks user phone number. pstnphone : The originating PSTN user phone number. userId : The originating BroadWorks user Id. url : The originating url. destination: The called user, number or URI. option : Additional vtr options. contact : The contact URL to use for the test call. diversion : The diversion URL to use for the test call. ========================================================================== vtr <origType>, Choice = {linePort, bwphone, pstnphone, userId, url} <linePort>, String {1 to 80 characters} <bwphone>, String {1 to 17 characters} <pstnphone>, String {1 to 17 characters} <userId>, String {2 to 161 characters} <url>, String {2 to 161 characters} <destination>, String {1 to 161 characters} [<option>, Multiple Choice = {contact, diversion}] <contact>, String {1 to 80 characters} <diversion>, String {1 to 80 characters} # Usage example: AS_CLI/ASDiagnostic/Diag> vtr userId ngnuserB1@mleus.lab 2012
```

Run VTR(I) Tool from the Web GUI

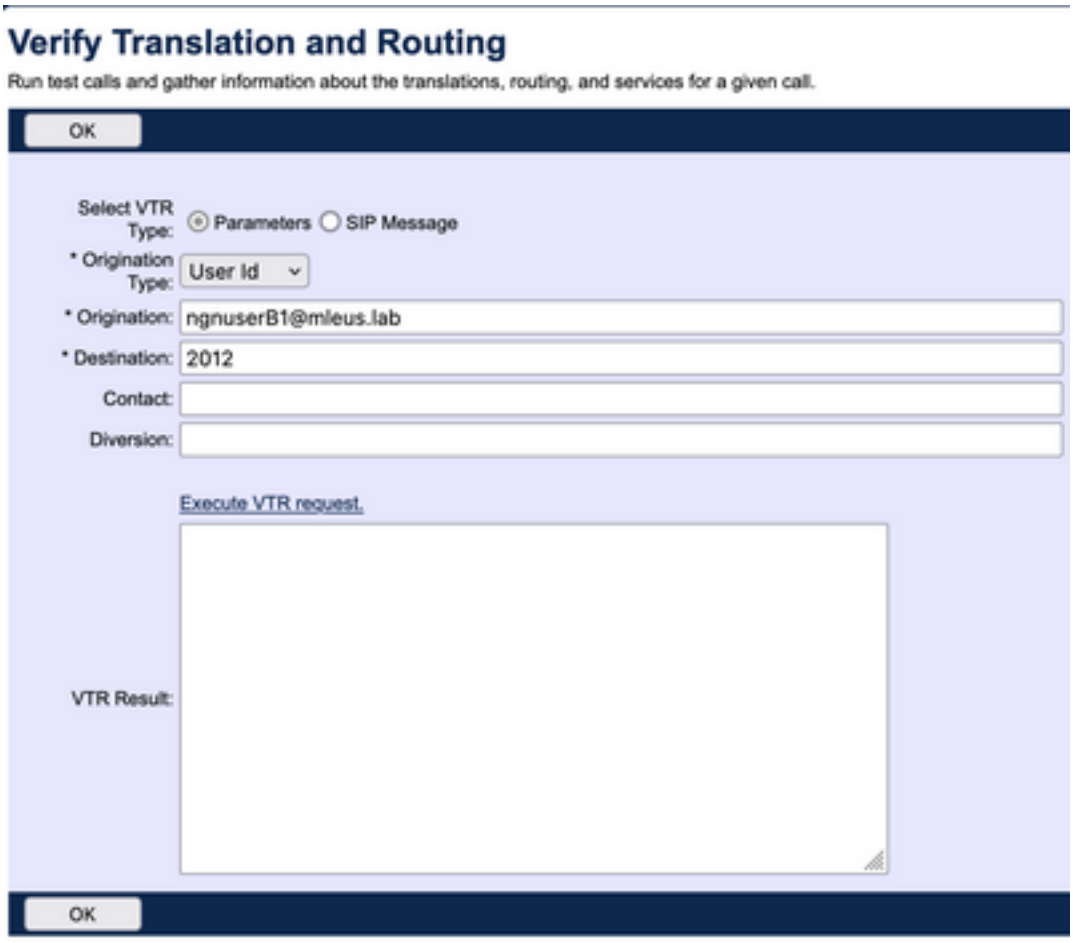
In order to run VTR from the Web GUI, use these steps:

Step 1. Navigate to **https://*<AS_FQDN>*/Login** page and log into the AS Web interface.

Step 2. Navigate to **System > Utilities > Verify Translation and Routing**.

Step 3. In order to use VTRI, click the **SIP Message** radio button and paste the SIP INVITE message in the input field. Then, click the **Execute VTR request** link:

Step 4. In order to use VTR, click the **Parameters** radio button and populate all required fields. Then click the **Execute VTR request** link:



VTR(I) Output Breakdown

Format of VTR and VTRI results is similar. It consists of eight sections:

1. Parameters or SIP INVITE (only present in VTR; not needed in VTRI because it works on real SIP message).
2. Originator information.
3. Originating call information.
4. Originating translation result.
5. Terminating call information.
6. Treatment Information
7. Timestamp.
8. Result.

Note: All the VTR outputs in this document are based on the **vtr userId *<user_id>* *<destination_number>*** command output, unless mentioned otherwise.

Parameters or SIP Invite

The first part of the VTR result is SIP INVITE, or the parameters used to simulate the call. This section is not visible if you provide SIP INVITE explicitly (in other words, if you use VTRI).

A SIP INVITE is generated when the origination is set to a URL or Line/Port. For example:

```
AS_CLI/ASDiagnostic/Diag> vtr linePort 5403362011@mleus.lab 2012 VTR Result: ------------------
---------------------------------- Using following SIP INVITE to run VTRI command with
Lineport ------------------------------------------------------ INVITE sip:2012@10.48.93.126
SIP/2.0 Via:SIP/2.0/UDP 127.0.0.1:5061;branch=vtr-unique-via-branch-26 From:"VTR Calling
Name"<sip:5403362011@mleus.lab>;tag=26 To:"VTR Called Name"<sip:2012@10.48.93.126> Call-ID:26
CSeq:26 INVITE Contact:<sip:5403362011@127.0.0.1:5061>
Allow:ACK,BYE,CANCEL,INFO,INVITE,OPTIONS,PRACK,REGISTER,UPDATE Content-Type:application/sdp
Content-Length:410 v=0 o=- 123 123 IN IP4 127.0.0.1 s=- c=IN IP4 127.0.0.1 t=0 0 m=audio 16428
RTP/AVP 0 2 4 8 18 96 97 98 100 101 a=rtpmap:0 PCMU/8000 a=rtpmap:2 G726-32/8000 a=rtpmap:4
G723/8000 a=rtpmap:8 PCMA/8000 a=rtpmap:18 G729a/8000 a=rtpmap:96 G726-40/8000 a=rtpmap:97 G726-
24/8000 a=rtpmap:98 G726-16/8000 a=rtpmap:100 NSE/8000 a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15 a=ptime:20 a=sendrecv
```

Parameters are shown if the origination is set to the User ID or Phone (both bwphone and pstnphone). For example:

```
AS_CLI/ASDiagnostic/Diag> vtr userId ngnuserB1@mleus.lab 2012 VTR Result: ----------------------
--------------------------------- Using following parameters to run short form VTR command ----
------------------------------------------------- VtrOriginationEvent vtrKey 16 origUserId
ngnuserB1@mleus.lab origUserUid 110439218 requestURI equivalent 2012@10.48.93.126:5060
dialedDigits (initial) 2012 deviceEnpoint 5403362011@mleus.lab
```

Originator Information

This section shows information about the originating party (caller). If the caller is a local BroadWorks user, detailed information about the user is shown. For example:

```
================================================== =========== ORIGINATOR INFO
================== ========================================================= [Orig-Id] VTR Short
form trigger. [Orig-Id] No Endpoint. [Orig-Id] Originating user type: BroadWorks [Orig-Id] User
Info [Orig-Id] User Id = ngnuserB1@mleus.lab [Orig-Id] User Uid = 110439218 [Orig-Id] Group Id =
ngngroupB1 [Orig-Id] Service Provider Id = ngnentB1 [Orig-Id] Reseller Id = null [Orig-Id] ASCII
First Name = Marek [Orig-Id] ASCII Last Name = Leus [Orig-Id] Unicode First Name = Marek [Orig-
Id] Unicode Last Name = Leus [Orig-Id] Country Code = 1 [Orig-Id] User Type = User [Orig-Id]
Trunk User Type = BroadWorks Regular User [Orig-Id] (0) Address type = main [Orig-Id] (0) dn =
+15403362011 [Orig-Id] (0) extension = 2011 [Orig-Id] activeAsComponentID = [vnf=null,vnfc=1]
[Orig-Id] beingRemoved = false [Orig-Id] configurable CLID = 5403362010 [Orig-Id]
synchronizationASRSentToNS = false
```

Originating Call Information

In this section, the progression of events on the caller side of the call is listed in order. If the caller is a Cisco BroadWorks user, then this is a place where the assigned services are executed. For example:

```
================================================== =========== ORIGINATING CALL INFO
================ ========================================================= [Orig/CallServiceBus]
CallId is callhalf-60507:0 [Orig/CallServiceBus] === Routing InvitationEvent on the Originating
Call bus === [Orig/CallServiceBus] TranslationServiceOrigInstance has CONSUMED the event.
[Orig/CallServiceBus] CallId is callhalf-60507:0 [Orig/CallServiceBus] === Routing
InvitationEvent on the Originating Call bus === [Orig/CallServiceBus] Resuming event processing
after TranslationServiceOrigInstance [Orig/CallServiceBus] SecurityClassificationServiceInstance
has processed the event...continue (...) # Output omitted for clarity (...)
[Orig/CallServiceBus] EmergencyCallTimerServiceInstance has processed the event...continue
[Orig/ReleaseWithCauseHandler] Post Processing RWC for Call.
```

Originating Translation Result

Information about a translation requested from the Cisco BroadWorks Network Server is provided in this section. If a network translation is required, information is shown about the used dial plan policy. It shows also the SIP messages exchanged for this purpose. For example:

```
================================================== =========== ORIGINATING TRANSLATION RESULT
========= ========================================================= [Orig-Xlation/DialPlanPolicy] --
Dial Plan Policy Information-- [Orig-Xlation/DialPlanPolicy] requiresAccessCodeForPublicCalls =
false [Orig-Xlation/DialPlanPolicy] allowE164PublicCalls = false [Orig-Xlation/DialPlanPolicy]
privateDigitMap = [Orig-Xlation/DialPlanPolicy] publicDigitMap = ([2-9]11|[0-1][2-
9]11|0[#T]|00|01[2-9]xx.[#T]|*xx|011x.[#T]|[0-1]xxxxxx[#T]|[0-1][2-9]xxxxxxxxx|[2-
9]xxxxxxxxx|[2-9]xxxxxx[#T]|101xxxx.[#T]|11|[2-9][#T]) [Orig-Xlation/DialPlanPolicy]
preferE164FormatForCallbackSvcs = false [Orig-Xlation/NetworkUsagePolicy] Network Usage Policy
is - do not force all calls to network - [Orig-Xlation/NetworkServerINVITE] Sending INVITE event
to network server for Translation Service Originating Side client [Orig-
Xlation/NetworkServerINVITE] Endpoint Id: callhalf-60487:0 udp 1128 SIP Bytes OUT to
10.48.93.128:5060 INVITE sip:2014@10.48.93.128;user=phone;transport=udp SIP/2.0 Via:SIP/2.0/UDP
10.48.93.126;branch=z9hG4bKBroadWorks.-iom24c-10.48.93.128V5060-0-960997887-1152157837-
1674811757564- From:"Marek Leus"<sip:+15403362011@10.48.93.126;user=phone>;tag=1152157837-
1674811757564- To:<sip:2014@10.48.93.128;user=phone> Call-
ID:BW102917564270123189259813 2@10.48.93.126 CSeq:960997887 INVITE
Contact:<sip:10.48.93.126:5060> P-Asserted-Identity:"Marek
Leus"<sip:+15403362011@10.48.93.126;user=phone> Privacy:none X-BroadWorks-Correlation-
Info:be8fde53-9ba5-4fc4-a788-ee426ed1fe90
Allow:ACK,BYE,CANCEL,INFO,INVITE,OPTIONS,PRACK,REFER,NOTIFY Supported: Max-Forwards:10 Content-
Type:application/sdp Content-Length:410 v=0 o=- 123 123 IN IP4 127.0.0.1 s=- c=IN IP4 127.0.0.1
```

```
t=0 0 m=audio 16428 RTP/AVP 0 2 4 8 18 96 97 98 100 101 a=rtpmap:0 PCMU/8000 a=rtpmap:2 G726-
32/8000 a=rtpmap:4 G723/8000 a=rtpmap:8 PCMA/8000 a=rtpmap:18 G729a/8000 a=rtpmap:96 G726-
40/8000 a=rtpmap:97 G726-24/8000 a=rtpmap:98 G726-16/8000 a=rtpmap:100 NSE/8000 a=rtpmap:101
telephone-event/8000 a=fmtp:101 0-15 a=ptime:20 a=sendrecv [Orig-Xlation/NS-RESPONSE] Got
Network Server response for Translation Service Originating Side client udp 394 SIP Bytes IN
from 10.48.93.128:49109 SIP/2.0 404 Not found Via:SIP/2.0/UDP
10.48.93.126;branch=z9hG4bKBroadWorks.-iom24c-10.48.93.128V5060-0-960997887-1152157837-
1674811757564- From:"Marek Leus"<sip:+15403362011@10.48.93.126;user=phone>;tag=1152157837-
1674811757564- To:<sip:2014@10.48.93.128;user=phone>;tag=1006595920-1674811749882 Call-
ID:BW102917564270123189259813 2@10.48.93.126 CSeq:960997887 INVITE Content-Length:0 [Orig-
Xlation/TranslationManager] Translation Client: Translation Service Originating Side call Id is
callhalf-60487:0 [Orig-Xlation/TranslationManager] === TranslationResult === [Orig-
Xlation/TranslationManager] callType Network [Orig-Xlation/TranslationManager] address 2014
[Orig-Xlation/TranslationManager] addressType main [Orig-Xlation/TranslationManager]
isServiceCode false [Orig-Xlation/TranslationManager] sc8Translated false [Orig-
Xlation/TranslationManager] sc100Translated false [Orig-Xlation/TranslationManager]
oacTranslated false [Orig-Xlation/TranslationManager] carrierPrefixTranslated false [Orig-
Xlation/TranslationManager] intraSP false [Orig-Xlation/TranslationManager] alias false [Orig-
Xlation/TranslationManager] preExtEmergencyRtgAK null [Orig-Xlation/TranslationManager] ===
Carrier Info === [Orig-Xlation/TranslationManager] ;csel=noind
```

Terminating Call Information

If the call has a valid destination, information is shown about its identity and the events on the terminating service bus. Similar to the originating call information section, this is a place where the assigned services of the destination user are executed. It also shows the SIP INVITE that is sent to the destination. For example:

```
================================================== =========== TERMINATING CALL INFO
=============== ======================================================
[Term/CallManagerServiceBus] CallManagerId is callhalf-60459 [Term/CallManagerServiceBus] ===
Routing TerminationEvent on the Call Manager bus === [Term/Term-Id] Terminating user type:
BroadWorks [Term/Term-Id] User Info [Term/Term-Id] User Id = ngnuserB2@mleus.lab [Term/Term-Id]
User Uid = 156778964 [Term/Term-Id] Group Id = ngngroupB1 [Term/Term-Id] Service Provider Id =
ngnentB1 [Term/Term-Id] Reseller Id = null [Term/Term-Id] ASCII First Name = John [Term/Term-Id]
ASCII Last Name = Doe [Term/Term-Id] Unicode First Name = John [Term/Term-Id] Unicode Last Name
= Doe [Term/Term-Id] Country Code = 1 [Term/Term-Id] User Type = User [Term/Term-Id] Trunk User
Type = BroadWorks Regular User [Term/Term-Id] (0) Address type = main [Term/Term-Id] (0) dn =
+15403362012 [Term/Term-Id] (0) extension = 2012 [Term/Term-Id] activeAsComponentID =
[vnf=null,vnfc=1] [Term/Term-Id] beingRemoved = false [Term/Term-Id] configurable CLID =
5403362010 [Term/Term-Id] synchronizationASRSentToNS = false [Term/CallServiceBus] CallId is
callhalf-60459:0 [Term/CallServiceBus] === Routing InvitationEvent on the Terminating Call bus
=== [Term/CallServiceBus] TranslationServiceTermInstance has processed the event...continue
[Term/CallServiceBus] NumberPortabilityQueryServiceTermInstance has processed the
event...continue (...) # Output omitted for clarity (...) [Term/CallManagerServiceBus]
RingTimeoutServiceInstance has processed the event...continue [Term/SipINVITE] Outgoing
resulting INVITE for Endpoint Id: callhalf-60459:0 udp 1195 SIP Bytes OUT to 10.61.71.93:62388
INVITE sip:5403362012@10.61.71.93:62388;transport=UDP;rinstance=c71abb15b3ff5d4a SIP/2.0
Via:SIP/2.0/UDP 10.48.93.126;branch=z9hG4bKBroadWorks.-iom24c-10.61.71.93V62388-0-960978525-
1694279432-1674811718841- From:"Marek Leus"<sip:2011@10.48.93.126;user=phone>;tag=1694279432-
1674811718841- To:"John Doe"<sip:5403362012@mleus.lab;rinstance=c71abb15b3ff5d4a> Call-
ID:BW102838841270123-1135433227@10.48.93.126 CSeq:960978525 INVITE
Contact:<sip:10.48.93.126:5060> Supported:100rel
Allow:ACK,BYE,CANCEL,INFO,INVITE,OPTIONS,PRACK,REFER,NOTIFY,UPDATE Call-
Info:<sip:10.48.93.126>;appearance-index=1 Recv-Info:x-broadworks-client-session-info
Accept:application/media_control+xml,application/sdp,multipart/mixed Max-Forwards:10 Content-
Type:application/sdp Content-Length:410 v=0 o=- 123 123 IN IP4 127.0.0.1 s=- c=IN IP4 127.0.0.1
t=0 0 m=audio 16428 RTP/AVP 0 2 4 8 18 96 97 98 100 101 a=rtpmap:0 PCMU/8000 a=rtpmap:2 G726-
32/8000 a=rtpmap:4 G723/8000 a=rtpmap:8 PCMA/8000 a=rtpmap:18 G729a/8000 a=rtpmap:96 G726-
40/8000 a=rtpmap:97 G726-24/8000 a=rtpmap:98 G726-16/8000 a=rtpmap:100 NSE/8000 a=rtpmap:101
```

```
telephone-event/8000 a=fmtp:101 0-15 a=ptime:20 a=sendrecv
```

Treatment Information

This section is present only for unsuccessful calls. It gives a detailed reason for the call failure. It also lists what announcement/tone is played by the Media Server if the call fails for that reason. For example:

```
================================================== ========== TREATMENT INFORMATION
========= ================================================== [Treatment/TreatmentsService]
Treatment service processing RWC. Release Cause is FORBIDDEN [Treatment/TreatmentsService]
defaultAnnouncement isTrtCallFailure.wav [Treatment/TreatmentsService] defaultTone is reorder-
pcmu.wav [Treatment/TreatmentsService] Configurable Treatment NOT used but specific audio
treatment files have been provided [Treatment/TreatmentsService] audioAnnouncement(s):
[Treatment/TreatmentsService] https://10.48.93.126/media/en/TrtSelectCallReject.wav
[Treatment/TreatmentsService] https://10.48.93.126/media/en/silence10s.wav
[Treatment/TreatmentsService] https://10.48.93.126/media/en/TrtSelectCallReject.wav
```

Timestamps

Call timestamps can be used to correlate events in VTRI with XSLogs.

Result

The final result of the test call, The possible values are:

- Allowed – The call can be successfully extended to the remote party. The outgoing INVITE is displayed in the Terminating Call Info section.
- Blocked – The call setup was blocked due to some policy or abnormal condition. The reason can be either on the origination side or on the termination side (dependent on the reason).
- Redirected – The call was redirected without a condition. The redirect destination is identified and shown in the report.
- Timeout – Something prevented the test call from terminating correctly and the 10-second internal time limit was reached. The information at hand can be used, but the final result is not shown.
- Media – A media component was started internally (usually an Interactive Voice Response [IVR]) to play a specific announcement or collect digits. Test calls are stopped in this case but the report shows the events up to this point.

Network Topology and Call Flow

For simplicity, a basic call scenario is used in this document:

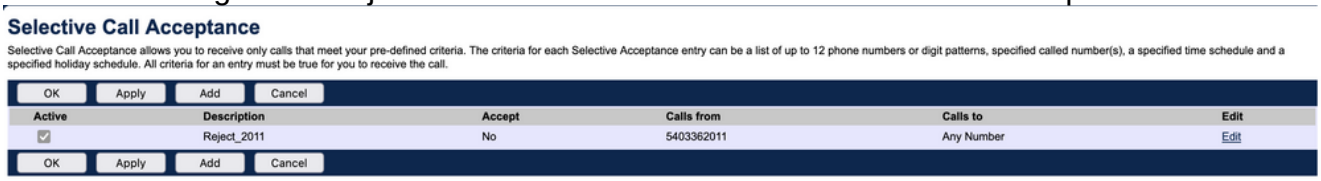
- Softphone applications are registered directly to AS.
- All the users are in the same group.

- There is neither call forward nor redirect used.

Call Scenarios

This document covers three scenarios where a user with extension 2011 dials in sequence:

1. Extension 2012, which is assigned to a registered device.
2. Extension 2012, which is assigned to a registered device, but the Selective Call Acceptance service is configured to reject the call from 2011 as shown in this screen capture:



3. Extension 2014, which is not assigned to any user.

Scenario 1: Successful call

In this scenario, one local user dials another local user by their extension. Let us go through the VTR output for this call to learn more about it:

```
AS_CLI/ASDiagnostic/Diag> vtr userId ngnuserB1@mleus.lab 2012 VTR Result: ----------------------
--------------------------------- Using following parameters to run short form VTR command ----
-------------------------------------------------- VtrOriginationEvent vtrKey 12 origUserId
ngnuserB1@mleus.lab origUserUid 110439218 requestURI equivalent 2012@10.48.93.126:5060
dialedDigits (initial) 2012 deviceEnpoint 5403362011@mleus.lab
```

In first section of the VTR results, you can see how AS maps parameters from the **vtr** command to SIP URLs:

- Caller: 5403362011@mleus.lab
- Destination: 2012@10.48.93.126:5060

```
================================================== =========== ORIGINATOR INFO
================== ========================================================= [Orig-Id] VTR Short
form trigger. [Orig-Id] No Endpoint. [Orig-Id] Originating user type: BroadWorks [Orig-Id] User
Info [Orig-Id] User Id = ngnuserB1@mleus.lab [Orig-Id] User Uid = 110439218 [Orig-Id] Group Id =
ngngroupB1 [Orig-Id] Service Provider Id = ngnentB1 [Orig-Id] Reseller Id = null [Orig-Id] ASCII
First Name = Marek [Orig-Id] ASCII Last Name = Leus [Orig-Id] Unicode First Name = Marek [Orig-
Id] Unicode Last Name = Leus [Orig-Id] Country Code = 1 [Orig-Id] User Type = User [Orig-Id]
Trunk User Type = BroadWorks Regular User [Orig-Id] (0) Address type = main [Orig-Id] (0) dn =
+15403362011 [Orig-Id] (0) extension = 2011 [Orig-Id] activeAsComponentID = [vnf=null,vnfc=1]
[Orig-Id] beingRemoved = false [Orig-Id] configurable CLID = 5403362010 [Orig-Id]
synchronizationASRSentToNS = false
```

Then, because the caller is local, all the details about the user are printed. From this section, you can learn what the user first (Marek) and last (Leus) name are, what their extension (2011) is, as well as what the full number (+15403362011) are, and what enterprise/group (ngnentB1/ngngroupB1) they belong to.

```
================================================== =========== ORIGINATING CALL INFO
=============== ========================================================= [Orig/CallServiceBus]
CallId is callhalf-60455:0 [Orig/CallServiceBus] === Routing InvitationEvent on the Originating
Call bus === [Orig/CallServiceBus] TranslationServiceOrigInstance has CONSUMED the event.
[Orig/CallServiceBus] CallId is callhalf-60455:0 [Orig/CallServiceBus] === Routing
InvitationEvent on the Originating Call bus === [Orig/CallServiceBus] Resuming event processing
after TranslationServiceOrigInstance [Orig/CallServiceBus] SecurityClassificationServiceInstance
has processed the event...continue [Orig/CallServiceBus] CMServiceInstance has processed the
event...continue [Orig/CallServiceBus] ReturnCallServiceInstance has processed the
event...continue [Orig/CallServiceBus] LNDServiceInstance has processed the event...continue
[Orig/CallServiceBus] GETSServiceInstance has processed the event...continue
[Orig/CallServiceBus] EnhancedCallLogsServiceInstance has processed the event...continue
[Orig/CallServiceBus] LocationControlServiceInstance has processed the event...continue
[Orig/CallServiceBus/OCPOriginatorServiceInstance] OCPOriginatorServiceInstance has processed
the event...continue [Orig/CallServiceBus/OCPOriginatorServiceInstance] Validating origination.
Using call type Group [Orig/CallServiceBus/OCPOriginatorServiceInstance] Validation returned
disposition Allow - OCP call type: Group [Orig/CallServiceBus] ACRFACServiceInstance has
processed the event...continue [Orig/CallServiceBus] NPAnnouncementServiceInstance has processed
the event...continue [Orig/CallServiceBus] CallCenterAgentCallServiceInstance has processed the
event...continue [Orig/CallServiceBus] PTTOriginatorServiceInstance has processed the
event...continue [Orig/CallServiceBus] LegacyACBServiceInstance has processed the
event...continue [Orig/CallServiceBus] AutomaticCallbackServiceInstance has processed the
event...continue [Orig/CallServiceBus] TreatmentsServiceInstance has processed the
event...continue [Orig/CallServiceBus] CFAlwaysFACServiceInstance has processed the
event...continue [Orig/CallServiceBus] SCFFACServiceInstance has processed the event...continue
[Orig/CallServiceBus] CallWaitingFACServiceInstance has processed the event...continue
[Orig/CallServiceBus] VMServiceInstance has processed the event...continue [Orig/CallServiceBus]
AnswerTimerFACServiceInstance has processed the event...continue [Orig/CallServiceBus]
DNDFACServiceInstance has processed the event...continue [Orig/CallServiceBus]
CPCServiceInstance has processed the event...continue [Orig/CallServiceBus]
ClassmarkServiceInstance has processed the event...continue [Orig/CallServiceBus]
VAOCallServiceInstance has processed the event...continue [Orig/CallServiceBus]
AoCServiceInstance has processed the event...continue [Orig/CallServiceBus]
EmergencyCallTimerServiceInstance has processed the event...continue [Orig/CallServiceBus]
SCRFACServiceInstance has processed the event...continue [Orig/CallServiceBus]
ExecutiveAssistantOrigServiceInstance has processed the event...continue [Orig/VTR_FINAL]
Triggering report.
```

As it is a simple user with no services enabled, there is not much information in the Originating Call Information section. The AS checks one service after another, and eventually moves to the next section of the VTR. It also defines this call as a group call. If you want to correlate this VTR analysis with XSLogs, you can extract the callhalf ID (callhalf-60455) from this section, and then use it to find this call in the log files. For example:

```
bwadmin@as1.mleus.lab$ less XSLog2023.01.27-00.00.00.txt | grep callhalf-60455 2023.01.27
10:28:38:819 CET | Info | Sip | BCCT Worker #1 | 3104778 | +15403362011 | callhalf-60455 $ new
CHSS > +15403362011 > > callhalf-60455 2023.01.27 10:28:38:819 CET | FieldDebug |
StateReplication | BCCT Worker #1 | 3104779 | +15403362011 | callhalf-60455 2023.01.27
10:28:38:821 CET | Info | CallP | Call Half Input Adapter 5 | 3104780 | Call Manager |
+15403362011 | ngngroupB1 | callhalf-60455
```

The next section is Originating Translation Result:

```
================================================== =========== ORIGINATING TRANSLATION RESULT
========= ========================================================= [Orig-Xlation/DialPlanPolicy] --
Dial Plan Policy Information-- [Orig-Xlation/DialPlanPolicy] requiresAccessCodeForPublicCalls =
false [Orig-Xlation/DialPlanPolicy] allowE164PublicCalls = false [Orig-Xlation/DialPlanPolicy]
privateDigitMap = [Orig-Xlation/DialPlanPolicy] publicDigitMap = ([2-9]11|[0-1][2-
9]11|0[#T]|00|01[2-9]xx.[#T]|*xx|011x.[#T]|[0-1]xxxxxx[#T]|[0-1][2-9]xxxxxxxxx|[2-
9]xxxxxxxxx|[2-9]xxxxxx[#T]|101xxxx.[#T]|11|[2-9][#T]) [Orig-Xlation/DialPlanPolicy]
preferE164FormatForCallbackSvcs = false [Orig-Xlation/NetworkUsagePolicy] Network Usage Policy
```

```
is - do not force all calls to network - [Orig-Xlation/TranslationManager] Translation Client:
Translation Service Originating Side call Id is callhalf-60455:0 [Orig-
Xlation/TranslationManager] === TranslationResult === [Orig-Xlation/TranslationManager] callType
Group [Orig-Xlation/TranslationManager] address 2012 [Orig-Xlation/TranslationManager]
destination uid 156778964 [Orig-Xlation/TranslationManager] destination user Id
ngnuserB2@mleus.lab [Orig-Xlation/TranslationManager] addressType main [Orig-
Xlation/TranslationManager] originalAddress 2012 [Orig-Xlation/TranslationManager]
dgcAlternateAddress 2012 [Orig-Xlation/TranslationManager] isServiceCode false [Orig-
Xlation/TranslationManager] sc8Translated false [Orig-Xlation/TranslationManager]
sc100Translated false [Orig-Xlation/TranslationManager] oacTranslated false [Orig-
Xlation/TranslationManager] carrierPrefixTranslated false [Orig-Xlation/TranslationManager]
intraSP false [Orig-Xlation/TranslationManager] alias false [Orig-Xlation/TranslationManager]
preExtEmergencyRtgAK null
```

From this section, you can learn what dial plan is applied for a given call. For this call, the output informs it is a call within the group and it is directed to user ngnuserB2@mleus.lab. The next section tells us more about this user:

```
================================================== =========== TERMINATING CALL INFO
=============== ========================================================
[Term/CallManagerServiceBus] CallManagerId is callhalf-60459 [Term/CallManagerServiceBus] ===
Routing TerminationEvent on the Call Manager bus === [Term/Term-Id] Terminating user type:
BroadWorks [Term/Term-Id] User Info [Term/Term-Id] User Id = ngnuserB2@mleus.lab [Term/Term-Id]
User Uid = 156778964 [Term/Term-Id] Group Id = ngngroupB1 [Term/Term-Id] Service Provider Id =
ngnentB1 [Term/Term-Id] Reseller Id = null [Term/Term-Id] ASCII First Name = John [Term/Term-Id]
ASCII Last Name = Doe [Term/Term-Id] Unicode First Name = John [Term/Term-Id] Unicode Last Name
= Doe [Term/Term-Id] Country Code = 1 [Term/Term-Id] User Type = User [Term/Term-Id] Trunk User
Type = BroadWorks Regular User [Term/Term-Id] (0) Address type = main [Term/Term-Id] (0) dn =
+15403362012 [Term/Term-Id] (0) extension = 2012 [Term/Term-Id] activeAsComponentID =
[vnf=null,vnfc=1] [Term/Term-Id] beingRemoved = false [Term/Term-Id] configurable CLID =
5403362010 [Term/Term-Id] synchronizationASRSentToNS = false
```

From this part, you learn same details about the call target as previously for caller: first and last name (John Doe), extension (2012), and full number (+15403362012), as well as the enterprise/group (ngnentB1/ngngroupB1).

```
[Term/CallServiceBus] CallId is callhalf-60459:0 [Term/CallServiceBus] === Routing
InvitationEvent on the Terminating Call bus === [Term/CallServiceBus]
TranslationServiceTermInstance has processed the event...continue [Term/CallServiceBus]
NumberPortabilityQueryServiceTermInstance has processed the event...continue
[Term/CallServiceBus] CMServiceInstance has processed the event...continue [Term/CallServiceBus]
GETSServiceInstance has processed the event...continue [Term/CallServiceBus]
CallCenterAgentCallServiceInstance has processed the event...continue [Term/CallServiceBus]
OCPTerminatorServiceInstance has processed the event...continue [Term/CallServiceBus]
ICPServiceInstance has processed the event...continue [Term/CallServiceBus] ACRServiceInstance
has processed the event...continue [Term/CallServiceBus/SCRServiceInstance] Criteria Set(s)
evaluated but not triggered: [Term/CallServiceBus/SCRServiceInstance]
[Term/CallServiceBus/SCRServiceInstance] Criteria Set: [Term/CallServiceBus/SCRServiceInstance]
description: Reject_all [Term/CallServiceBus/SCRServiceInstance] active: false
[Term/CallServiceBus/SCRServiceInstance] blackListFlag: false
[Term/CallServiceBus/SCRServiceInstance] ScheduleCriteria
[Term/CallServiceBus/SCRServiceInstance] - No Schedule associated
[Term/CallServiceBus/SCRServiceInstance] - inScheduleRightNow: true
[Term/CallServiceBus/SCRServiceInstance] CallingPartyAddressCriteria
[Term/CallServiceBus/SCRServiceInstance] Any Address [Term/CallServiceBus/SCRServiceInstance]
CallToCriteria [Term/CallServiceBus/SCRServiceInstance] CallTo matches ANY [Term/CallServiceBus]
SCRServiceInstance has processed the event...continue [Term/CallServiceBus]
PTTTerminatorServiceInstance has processed the event...continue [Term/CallServiceBus]
LNRServiceInstance has processed the event...continue [Term/CallServiceBus]
EnhancedCallLogsServiceInstance has processed the event...continue [Term/CallServiceBus]
GroupNightForwardingServiceInstance has processed the event...continue [Term/CallServiceBus]
```

```
FaxMessagingServiceInstance has processed the event...continue [Term/CallServiceBus]
SCFServiceInstance has processed the event...continue [Term/CallServiceBus]
CFAlwaysTerminatorServiceInstance has processed the event...continue [Term/CallServiceBus]
DNDServiceInstance has processed the event...continue [Term/CallServiceBus]
PersonalAssistantServiceInstance has processed the event...continue [Term/CallServiceBus]
CallWaitingTerminatorServiceInstance has processed the event...continue [Term/CallServiceBus]
PreAlertingAnnouncementServiceInstance has processed the event...continue [Term/CallServiceBus]
AnswerTimerServiceInstance has processed the event...continue [Term/CallServiceBus]
VMServiceInstance has processed the event...continue [Term/CallServiceBus]
RedirectionServiceInstance has processed the event...continue [Term/CallServiceBus]
SecurityClassificationServiceInstance has processed the event...continue [Term/CallServiceBus]
ExecutiveTermServiceInstance has processed the event...continue [Term/CallServiceBus]
ExecutiveAssistantTermServiceInstance has processed the event...continue
```

Further in this section, all services are checked. There is only one service assigned to this user (Selective Call Acceptance), but the only Criteria Set (Reject_2011) is inactive. Therefore, it is not triggered and VTR analysis moves to the next stage of the Terminating Call Info section, which shows the final result of analysis.

```
[Term/CallManagerServiceBus] CallManagerId is callhalf-60459 [Term/CallManagerServiceBus] ===
Routing InvitationEvent on the Call Manager bus === [Term/CallManagerServiceBus]
AccessRoutingServiceInstance has processed the event...continue [Term/CallManagerServiceBus]
VAOServiceInstance has processed the event...continue [Term/CallManagerServiceBus]
TSDServiceInstance has processed the event...continue [Term/CallManagerServiceBus]
SCAPServiceInstance has processed the event...continue [Term/CallManagerServiceBus]
CallCenterAgentServiceInstance has processed the event...continue [Term/CallManagerServiceBus]
FlashServiceInstance has processed the event...continue [Term/CallManagerServiceBus]
RingTimeoutServiceInstance has processed the event...continue [Term/SipINVITE] Outgoing
resulting INVITE for Endpoint Id: callhalf-60459:0 udp 1195 SIP Bytes OUT to 10.61.71.93:62388
INVITE sip:5403362012@10.61.71.93:62388;transport=UDP;rinstance=c71abb15b3ff5d4a SIP/2.0
Via:SIP/2.0/UDP 10.48.93.126;branch=z9hG4bKBroadWorks.-iom24c-10.61.71.93V62388-0-960978525-
1694279432-1674811718841- From:"Marek Leus"<sip:2011@10.48.93.126;user=phone>;tag=1694279432-
1674811718841- To:"John Doe"<sip:5403362012@mleus.lab;rinstance=c71abb15b3ff5d4a> Call-
ID:BW102838841270123-1135433227@10.48.93.126 CSeq:960978525 INVITE
Contact:<sip:10.48.93.126:5060> Supported:100rel
Allow:ACK,BYE,CANCEL,INFO,INVITE,OPTIONS,PRACK,REFER,NOTIFY,UPDATE Call-
Info:<sip:10.48.93.126>;appearance-index=1 Recv-Info:x-broadworks-client-session-info
Accept:application/media_control+xml,application/sdp,multipart/mixed Max-Forwards:10 Content-
Type:application/sdp Content-Length:410 v=0 o=- 123 123 IN IP4 127.0.0.1 s=- c=IN IP4 127.0.0.1
t=0 0 m=audio 16428 RTP/AVP 0 2 4 8 18 96 97 98 100 101 a=rtpmap:0 PCMU/8000 a=rtpmap:2 G726-
32/8000 a=rtpmap:4 G723/8000 a=rtpmap:8 PCMA/8000 a=rtpmap:18 G729a/8000 a=rtpmap:96 G726-
40/8000 a=rtpmap:97 G726-24/8000 a=rtpmap:98 G726-16/8000 a=rtpmap:100 NSE/8000 a=rtpmap:101
telephone-event/8000 a=fmtp:101 0-15 a=ptime:20 a=sendrecv
```

From this section, you see the INVITE message that would be extended to the final destination if it was a real call. You can also extract callhalf ID (callhalf-60459) that can be used to trace this call in the log files:

```
bwadmin@as1.mleus.lab$ less XSLog2023.01.27-00.00.00.txt | grep callhalf-60459 2023.01.27
10:28:38:827 CET | Info | Sip | Call Half Input Adapter 5 | 3104796 | +15403362012 | callhalf-
60459 | 5ff0de06-90a9-4bb6-892a-610d7441ed69 $ new CHSS > +15403362012 > callhalf-60455:0 >
callhalf-60459 2023.01.27 10:28:38:827 CET | FieldDebug | StateReplication | Call Half Input
Adapter 5 | 3104797 | +15403362012 | callhalf-60459 | 5ff0de06-90a9-4bb6-892a-610d7441ed69
2023.01.27 10:28:38:827 CET | Info | CallP | Call Half Input Adapter 0 | 3104798 | Call Manager
| +15403362012 | ngngroupB1 | callhalf-60459 | 5ff0de06-90a9-4bb6-892a-610d7441ed69
```

The last two sections show the final result and timestamps of this analysis:

```
================================================== ========== TIMESTAMPS =========
================================================== [Timestamps] Start time = 2023.01.27
```

```
10:28:38:819 CET [Timestamps] End time = 2023.01.27 10:28:38:846 CET
================================================== ========== RESULT: ALLOWED
============== ==================================================
```

Scenario 2: Call Rejected by the Selective Call Acceptance Service

In this scenario, one local user again dials another local user by their extension; however, this time the Selective Call Acceptance is configured to reject this call. Let us go through the VTR output for this call to learn more about it:

```
AS_CLI/ASDiagnostic/Diag> vtr userId ngnuserB1@mleus.lab 2012 VTR Result: ----------------------
--------------------------------- Using following parameters to run short form VTR command ----
-------------------------------------------------- VtrOriginationEvent vtrKey 16 origUserId
ngnuserB1@mleus.lab origUserUid 110439218 requestURI equivalent 2012@10.48.93.126:5060
dialedDigits (initial) 2012 deviceEnpoint 5403362011@mleus.lab
```

In the first section of the VTR results, you can see how the AS maps parameters from the **vtr** command to SIP URLs:

- Caller: 5403362011@mleus.lab
- Destination: 2012@10.48.93.126:5060

Because it is exactly the same caller as in the first scenario, the Originator Information section for this is exactly the same as well. Therefore, it is skipped for clarity and analysis continues to the Originating Call Information section:

```
================================================== ========== ORIGINATING CALL INFO
=============== ================================================== [Orig/CallServiceBus]
CallId is callhalf-60507:0 [Orig/CallServiceBus] === Routing InvitationEvent on the Originating
Call bus === [Orig/CallServiceBus] TranslationServiceOrigInstance has CONSUMED the event.
[Orig/CallServiceBus] CallId is callhalf-60507:0 [Orig/CallServiceBus] === Routing
InvitationEvent on the Originating Call bus === [Orig/CallServiceBus] Resuming event processing
after TranslationServiceOrigInstance [Orig/CallServiceBus] SecurityClassificationServiceInstance
has processed the event...continue [Orig/CallServiceBus] CMServiceInstance has processed the
event...continue [Orig/CallServiceBus] ReturnCallServiceInstance has processed the
event...continue [Orig/CallServiceBus] LNDServiceInstance has processed the event...continue
[Orig/CallServiceBus] GETSServiceInstance has processed the event...continue
[Orig/CallServiceBus] EnhancedCallLogsServiceInstance has processed the event...continue
[Orig/CallServiceBus] LocationControlServiceInstance has processed the event...continue
[Orig/CallServiceBus/OCPOriginatorServiceInstance] OCPOriginatorServiceInstance has processed
the event...continue [Orig/CallServiceBus/OCPOriginatorServiceInstance] Validating origination.
Using call type Group [Orig/CallServiceBus/OCPOriginatorServiceInstance] Validation returned
disposition Allow - OCP call type: Group [Orig/CallServiceBus] ACRFACServiceInstance has
processed the event...continue [Orig/CallServiceBus] NPAnnouncementServiceInstance has processed
the event...continue [Orig/CallServiceBus] CallCenterAgentCallServiceInstance has processed the
event...continue [Orig/CallServiceBus] PTTOriginatorServiceInstance has processed the
event...continue [Orig/CallServiceBus] LegacyACBServiceInstance has processed the
event...continue [Orig/CallServiceBus] AutomaticCallbackServiceInstance has processed the
event...continue [Orig/CallServiceBus] TreatmentsServiceInstance has processed the
event...continue [Orig/CallServiceBus] CFAlwaysFACServiceInstance has processed the
event...continue [Orig/CallServiceBus] SCFFACServiceInstance has processed the event...continue
[Orig/CallServiceBus] CallWaitingFACServiceInstance has processed the event...continue
[Orig/CallServiceBus] VMServiceInstance has processed the event...continue [Orig/CallServiceBus]
AnswerTimerFACServiceInstance has processed the event...continue [Orig/CallServiceBus]
DNDFACServiceInstance has processed the event...continue [Orig/CallServiceBus]
CPCServiceInstance has processed the event...continue [Orig/CallServiceBus]
```

```
ClassmarkServiceInstance has processed the event...continue [Orig/CallServiceBus]
VAOCallServiceInstance has processed the event...continue [Orig/CallServiceBus]
AoCServiceInstance has processed the event...continue [Orig/CallServiceBus]
EmergencyCallTimerServiceInstance has processed the event...continue [Orig/CallServiceBus]
SCRFACServiceInstance has processed the event...continue [Orig/CallServiceBus]
ExecutiveAssistantOrigServiceInstance has processed the event...continue [Orig/CallServiceBus]
CallId is callhalf-60507:0 [Orig/CallServiceBus] === Routing ReleaseWithCauseEvent on the
Originating Call bus === [Orig/CallServiceBus] TranslationServiceOrigInstance has processed the
event...continue [Orig/CallServiceBus] CallTransferRecallServiceInstance has processed the
event...continue [Orig/CallServiceBus] CMServiceInstance has processed the event...continue
[Orig/CallServiceBus] GETSServiceInstance has processed the event...continue
[Orig/CallServiceBus] EnhancedCallLogsServiceInstance has processed the event...continue
[Orig/CallServiceBus] OCPOriginatorServiceInstance has processed the event...continue
[Orig/CallServiceBus] CallCenterAgentCallServiceInstance has processed the event...continue
[Orig/CallServiceBus] PTTOriginatorServiceInstance has processed the event...continue
[Orig/CallServiceBus] CallParkServiceInstance has processed the event...continue
[Orig/CallServiceBus] LegacyACBServiceInstance has processed the event...continue
[Orig/CallServiceBus] AutomaticCallbackServiceInstance has processed the event...continue
[Orig/CallServiceBus/TreatmentsServiceInstance] TreatmentsServiceInstance has processed the
event...continue [Orig/CallServiceBus/TreatmentsServiceInstance] Treatments Service would have
started the announcement (see Treatment Section for more details).
[Orig/CallServiceBus/TreatmentsServiceInstance] Letting event continue to release test call
resources properly [Orig/CallServiceBus] CallWaitingFACServiceInstance has processed the
event...continue [Orig/CallServiceBus] CFBusyFACServiceInstance has processed the
event...continue [Orig/CallServiceBus] CFNoAnswerFACServiceInstance has processed the
event...continue [Orig/CallServiceBus] VMServiceInstance has processed the event...continue
[Orig/CallServiceBus] CFNotReachableFACServiceInstance has processed the event...continue
[Orig/CallServiceBus] DirectedCallPickupServiceInstance has processed the event...continue
[Orig/CallServiceBus] DPUBIServiceInstance has processed the event...continue
[Orig/CallServiceBus] AnswerTimerFACServiceInstance has processed the event...continue
[Orig/CallServiceBus] MusicOnHoldServiceInstance has processed the event...continue
[Orig/CallServiceBus] AoCServiceInstance has processed the event...continue
[Orig/CallServiceBus] EmergencyCallTimerServiceInstance has processed the event...continue
[Orig/ReleaseWithCauseHandler] Post Processing RWC for Call.
```

This section shows that the destination number is known to the AS and it defines the call as a group call. However, you can also determine if something goes wrong. First, there is the "Routing ReleaseWithCauseEvent on the Originating Call bus" statement and TreatmentServiceInstance is involved. Likewise, the section is concluded with ReleaseWithCauseHandler.

The next section is Originating Translation Result, which is identical with the section from Scenario 1 because the call is directed to the same user. Therefore, it is omitted for clarity. The difference appears in the Terminating Call Info section:

```
================================================== =========== TERMINATING CALL INFO
=============== ==================================================
[Term/CallManagerServiceBus] CallManagerId is callhalf-60511 [Term/CallManagerServiceBus] ===
Routing TerminationEvent on the Call Manager bus === [Term/Term-Id] Terminating user type:
BroadWorks [Term/Term-Id] User Info [Term/Term-Id] User Id = ngnuserB2@mleus.lab [Term/Term-Id]
User Uid = 156778964 [Term/Term-Id] Group Id = ngngroupB1 [Term/Term-Id] Service Provider Id =
ngnentB1 [Term/Term-Id] Reseller Id = null [Term/Term-Id] ASCII First Name = John [Term/Term-Id]
ASCII Last Name = Doe [Term/Term-Id] Unicode First Name = John [Term/Term-Id] Unicode Last Name
= Doe [Term/Term-Id] Country Code = 1 [Term/Term-Id] User Type = User [Term/Term-Id] Trunk User
Type = BroadWorks Regular User [Term/Term-Id] (0) Address type = main [Term/Term-Id] (0) dn =
+15403362012 [Term/Term-Id] (0) extension = 2012 [Term/Term-Id] activeAsComponentID =
[vnf=null,vnfc=1] [Term/Term-Id] beingRemoved = false [Term/Term-Id] configurable CLID =
5403362010 [Term/Term-Id] synchronizationASRSentToNS = false [Term/CallServiceBus] CallId is
callhalf-60511:0 [Term/CallServiceBus] === Routing InvitationEvent on the Terminating Call bus
=== [Term/CallServiceBus] TranslationServiceTermInstance has processed the event...continue
[Term/CallServiceBus] NumberPortabilityQueryServiceTermInstance has processed the
```

```
event...continue [Term/CallServiceBus] CMServiceInstance has processed the event...continue
[Term/CallServiceBus] GETSServiceInstance has processed the event...continue
[Term/CallServiceBus] CallCenterAgentCallServiceInstance has processed the event...continue
[Term/CallServiceBus] OCPTerminatorServiceInstance has processed the event...continue
[Term/CallServiceBus] ICPServiceInstance has processed the event...continue
[Term/CallServiceBus] ACRServiceInstance has processed the event...continue
[Term/CallServiceBus/SCAServiceInstance] Criteria Set evaluated and triggered:
[Term/CallServiceBus/SCAServiceInstance] [Term/CallServiceBus/SCAServiceInstance] Criteria Set:
[Term/CallServiceBus/SCAServiceInstance] description: Reject_2011
[Term/CallServiceBus/SCAServiceInstance] active: true [Term/CallServiceBus/SCAServiceInstance]
blackListFlag: true [Term/CallServiceBus/SCAServiceInstance] ScheduleCriteria
[Term/CallServiceBus/SCAServiceInstance] - No Schedule associated
[Term/CallServiceBus/SCAServiceInstance] - inScheduleRightNow: true
[Term/CallServiceBus/SCAServiceInstance] CallingPartyAddressCriteria
[Term/CallServiceBus/SCAServiceInstance] +15403362011 [Term/CallServiceBus/SCAServiceInstance]
CallToCriteria [Term/CallServiceBus/SCAServiceInstance] CallTo matches ANY [Term/CallServiceBus]
SCAServiceInstance has CONSUMED the event. [Term/CallServiceBus] CallId is callhalf-60511:0
[Term/CallServiceBus] === Routing ReleaseWithCauseEvent on the Terminating Call bus ===
[Term/CallServiceBus] TranslationServiceTermInstance has processed the event...continue
[Term/CallServiceBus] CMServiceInstance has processed the event...continue [Term/CallServiceBus]
CallParkServiceInstance has processed the event...continue [Term/CallServiceBus]
GETSServiceInstance has processed the event...continue [Term/CallServiceBus]
CallCenterAgentCallServiceInstance has processed the event...continue [Term/CallServiceBus]
OCPTerminatorServiceInstance has processed the event...continue [Term/CallServiceBus]
PTTTerminatorServiceInstance has processed the event...continue [Term/CallServiceBus]
LNRServiceInstance has processed the event...continue [Term/CallServiceBus]
EnhancedCallLogsServiceInstance has processed the event...continue [Term/CallServiceBus]
PersonalAssistantServiceInstance has processed the event...continue [Term/CallServiceBus]
CallWaitingTerminatorServiceInstance has processed the event...continue [Term/CallServiceBus]
CFBusyTerminatorServiceInstance has processed the event...continue [Term/CallServiceBus]
AnswerTimerServiceInstance has processed the event...continue [Term/CallServiceBus]
CFNotReachableTerminatorServiceInstance has processed the event...continue [Term/CallServiceBus]
CFNoAnswerTerminatorServiceInstance has processed the event...continue [Term/CallServiceBus]
VMServiceInstance has processed the event...continue [Term/CallServiceBus]
DirectedCallPickupServiceInstance has processed the event...continue [Term/CallServiceBus]
DPUBIServiceInstance has processed the event...continue [Term/CallServiceBus]
RedirectionServiceInstance has processed the event...continue [Term/CallServiceBus]
MusicOnHoldServiceInstance has processed the event...continue
```

You can see that this time the Reject_2011 criteria set is active. As result so Selective Call Acceptance service is triggered. As a result, ReleaseWithCauseEvent is raised on the Terminating Call bus. The AS checks the rest of services and moves to the Treatment Information section:

```
================================================== ========== TREATMENT INFORMATION
========= ================================================== [Treatment/TreatmentsService]
Treatment service processing RWC. Release Cause is FORBIDDEN [Treatment/TreatmentsService]
defaultAnnouncement isTrtCallFailure.wav [Treatment/TreatmentsService] defaultTone is reorder-
pcmu.wav [Treatment/TreatmentsService] Configurable Treatment NOT used but specific audio
treatment files have been provided [Treatment/TreatmentsService] audioAnnouncement(s):
[Treatment/TreatmentsService] https://10.48.93.126/media/en/TrtSelectCallReject.wav
[Treatment/TreatmentsService] https://10.48.93.126/media/en/silence10s.wav
[Treatment/TreatmentsService] https://10.48.93.126/media/en/TrtSelectCallReject.wav
```

From this section you can learn the reason of the call failure (FORBIDDEN) and that the TrtSelectCallReject.wav announcement is to be played.

The last two sections show the final result and timestamps of this analysis:

```
================================================== ========== TIMESTAMPS =========
================================================== [Timestamps] Start time = 2023.01.27
10:31:29:425 CET [Timestamps] End time = 2023.01.27 10:31:29:466 CET
```

```
================================================== ========== RESULT: BLOCKED
============== ==================================================
```

Scenario 3: Call to Unassigned Number

In this scenario, one local user dials another extension that is unassigned. Let us go through the VTR output for this call to learn more about it:

```
AS_CLI/ASDiagnostic/Diag> vtr userId ngnuserB1@mleus.lab 2014 VTR Result: ----------------------
--------------------------------- Using following parameters to run short form VTR command ----
------------------------------------------------- VtrOriginationEvent vtrKey 14 origUserId
ngnuserB1@mleus.lab origUserUid 110439218 requestURI equivalent 2014@10.48.93.126:5060
dialedDigits (initial) 2014 deviceEnpoint 5403362011@mleus.lab
```

In the first section of VTR results, you can see how the AS maps parameters from the **vtr** command to SIP URLs:

- Caller: 5403362011@mleus.lab
- Destination: 2014@10.48.93.126:5060

Because it is exactly the same caller as in the first scenario, the Originator Information section for this is exactly the same as well. Therefore, it is skipped for clarity and analysis continues to the Originating Call Information section:

```
================================================== ========== ORIGINATING CALL INFO
=============== ==================================================== [Orig/CallServiceBus]
CallId is callhalf-60487:0 [Orig/CallServiceBus] === Routing InvitationEvent on the Originating
Call bus === [Orig/CallServiceBus] TranslationServiceOrigInstance has CONSUMED the event.
[Orig/CallServiceBus] CallId is callhalf-60487:0 [Orig/CallServiceBus] === Routing
ReleaseWithCauseEvent on the Originating Call bus === [Orig/CallServiceBus]
TranslationServiceOrigInstance has processed the event...continue [Orig/CallServiceBus]
CallTransferRecallServiceInstance has processed the event...continue [Orig/CallServiceBus]
CMServiceInstance has processed the event...continue [Orig/CallServiceBus] GETSServiceInstance
has processed the event...continue [Orig/CallServiceBus] EnhancedCallLogsServiceInstance has
processed the event...continue [Orig/CallServiceBus] OCPOriginatorServiceInstance has processed
the event...continue [Orig/CallServiceBus] CallCenterAgentCallServiceInstance has processed the
event...continue [Orig/CallServiceBus] PTTOriginatorServiceInstance has processed the
event...continue [Orig/CallServiceBus] CallParkServiceInstance has processed the
event...continue [Orig/CallServiceBus] LegacyACBServiceInstance has processed the
event...continue [Orig/CallServiceBus] AutomaticCallbackServiceInstance has processed the
event...continue [Orig/CallServiceBus/TreatmentsServiceInstance] TreatmentsServiceInstance has
processed the event...continue [Orig/CallServiceBus/TreatmentsServiceInstance] Treatments
Service would have started the announcement (see Treatment Section for more details).
[Orig/CallServiceBus/TreatmentsServiceInstance] Letting event continue to release test call
resources properly [Orig/CallServiceBus] CallWaitingFACServiceInstance has processed the
event...continue [Orig/CallServiceBus] CFBusyFACServiceInstance has processed the
event...continue [Orig/CallServiceBus] CFNoAnswerFACServiceInstance has processed the
event...continue [Orig/CallServiceBus] VMServiceInstance has processed the event...continue
[Orig/CallServiceBus] CFNotReachableFACServiceInstance has processed the event...continue
[Orig/CallServiceBus] DirectedCallPickupServiceInstance has processed the event...continue
[Orig/CallServiceBus] DPUBIServiceInstance has processed the event...continue
[Orig/CallServiceBus] AnswerTimerFACServiceInstance has processed the event...continue
[Orig/CallServiceBus] MusicOnHoldServiceInstance has processed the event...continue
[Orig/CallServiceBus] AoCServiceInstance has processed the event...continue
[Orig/CallServiceBus] EmergencyCallTimerServiceInstance has processed the event...continue
[Orig/ReleaseWithCauseHandler] Post Processing RWC for Call.
```

From this section, you can already figure out that something is wrong. First, there is a "Routing ReleaseWithCauseEvent on the Originating Call bus" statement instead of the "Routing InvitationEvent on the Originating Call bus" statement. Then, there is a TreatmentServiceInstance statement involved. Finally, the section ends up with the ReleaseWithCauseHandler statement.

```
================================================== ========== ORIGINATING TRANSLATION RESULT
========= ================================================== [Orig-Xlation/DialPlanPolicy] --
Dial Plan Policy Information-- [Orig-Xlation/DialPlanPolicy] requiresAccessCodeForPublicCalls =
false [Orig-Xlation/DialPlanPolicy] allowE164PublicCalls = false [Orig-Xlation/DialPlanPolicy]
privateDigitMap = [Orig-Xlation/DialPlanPolicy] publicDigitMap = ([2-9]11|[0-1][2-
9]11|0[#T]|00|01[2-9]xx.[#T]|*xx|011x.[#T]|[0-1]xxxxxx[#T]|[0-1][2-9]xxxxxxxxx|[2-
9]xxxxxxxxx|[2-9]xxxxxx[#T]|101xxxx.[#T]|11|[2-9][#T]) [Orig-Xlation/DialPlanPolicy]
preferE164FormatForCallbackSvcs = false [Orig-Xlation/NetworkUsagePolicy] Network Usage Policy
is - do not force all calls to network - [Orig-Xlation/NetworkServerINVITE] Sending INVITE event
to network server for Translation Service Originating Side client [Orig-
Xlation/NetworkServerINVITE] Endpoint Id: callhalf-60487:0 udp 1128 SIP Bytes OUT to
10.48.93.128:5060 INVITE sip:2014@10.48.93.128;user=phone;transport=udp SIP/2.0 Via:SIP/2.0/UDP
10.48.93.126;branch=z9hG4bKBroadWorks.-iom24c-10.48.93.128V5060-0-960997887-1152157837-
1674811757564- From:"Marek Leus"<sip:+15403362011@10.48.93.126;user=phone>;tag=1152157837-
1674811757564- To:<sip:2014@10.48.93.128;user=phone> Call-
ID:BW102917564270123189259813 2@10.48.93.126 CSeq:960997887 INVITE
Contact:<sip:10.48.93.126:5060> P-Asserted-Identity:"Marek
Leus"<sip:+15403362011@10.48.93.126;user=phone> Privacy:none X-BroadWorks-Correlation-
Info:be8fde53-9ba5-4fc4-a788-ee426ed1fe90
Allow:ACK,BYE,CANCEL,INFO,INVITE,OPTIONS,PRACK,REFER,NOTIFY Supported: Max-Forwards:10 Content-
Type:application/sdp Content-Length:410 v=0 o=- 123 123 IN IP4 127.0.0.1 s=- c=IN IP4 127.0.0.1
t=0 0 m=audio 16428 RTP/AVP 0 2 4 8 18 96 97 98 100 101 a=rtpmap:0 PCMU/8000 a=rtpmap:2 G726-
32/8000 a=rtpmap:4 G723/8000 a=rtpmap:8 PCMA/8000 a=rtpmap:18 G729a/8000 a=rtpmap:96 G726-
40/8000 a=rtpmap:97 G726-24/8000 a=rtpmap:98 G726-16/8000 a=rtpmap:100 NSE/8000 a=rtpmap:101
telephone-event/8000 a=fmtp:101 0-15 a=ptime:20 a=sendrecv [Orig-Xlation/NS-RESPONSE] Got
Network Server response for Translation Service Originating Side client udp 394 SIP Bytes IN
from 10.48.93.128:49109 SIP/2.0 404 Not found Via:SIP/2.0/UDP
10.48.93.126;branch=z9hG4bKBroadWorks.-iom24c-10.48.93.128V5060-0-960997887-1152157837-
1674811757564- From:"Marek Leus"<sip:+15403362011@10.48.93.126;user=phone>;tag=1152157837-
1674811757564- To:<sip:2014@10.48.93.128;user=phone>;tag=1006595920-1674811749882 Call-
ID:BW102917564270123189259813 2@10.48.93.126 CSeq:960997887 INVITE Content-Length:0 [Orig-
Xlation/TranslationManager] Translation Client: Translation Service Originating Side call Id is
callhalf-60487:0 [Orig-Xlation/TranslationManager] === TranslationResult === [Orig-
Xlation/TranslationManager] callType Network [Orig-Xlation/TranslationManager] address 2014
[Orig-Xlation/TranslationManager] addressType main [Orig-Xlation/TranslationManager]
isServiceCode false [Orig-Xlation/TranslationManager] sc8Translated false [Orig-
Xlation/TranslationManager] sc100Translated false [Orig-Xlation/TranslationManager]
oacTranslated false [Orig-Xlation/TranslationManager] carrierPrefixTranslated false [Orig-
Xlation/TranslationManager] intraSP false [Orig-Xlation/TranslationManager] alias false [Orig-
Xlation/TranslationManager] preExtEmergencyRtgAK null [Orig-Xlation/TranslationManager] ===
Carrier Info === [Orig-Xlation/TranslationManager] ;csel=noind
```

The Originating Translation Result section gives an insight as to why the call is not successful. It is clear from the output that the AS tries to resolve the 2014 number with the NS, but it returns the 404 Not Found error response.

Because the call cannot be extended to its destination, the Terminating Call Info section is very brief:

```
================================================== ========== TERMINATING CALL INFO
=============== ================================================== [Term] Terminating call
processing information has not been populated
```

Instead, there is a Treatment Information section present with information on how the failed call must be handled:

```
================================================== =========== TREATMENT INFORMATION
========= ========================================================= [Treatment/TreatmentsService]
Treatment service processing RWC. Release Cause is USER_NOT_FOUND [Treatment/TreatmentsService]
defaultAnnouncement isTrtUnknownUser.wav [Treatment/TreatmentsService] defaultTone is reorder-
pcmu.wav [Treatment/TreatmentsService] Treatment playing default announcement
https://10.48.93.126/media/en/TrtUnknownUser.wav
```

This sections gives the reason for the call failure (USER_NOT_FOUND) and informs that TrtUnknownUser.wav announcement and/or reorder-pcmu.wav tone is to be played. It also points the path to be used in dialog with the Media Server to instruct it to play the proper announcement.

The last two sections show the final result and timestamps of this analysis:

```
================================================== =========== TIMESTAMPS =========
================================================== [Timestamps] Start time = 2023.01.27
10:29:17:525 CET [Timestamps] End time = 2023.01.27 10:29:17:669 CET
================================================== =========== RESULT: BLOCKED
=============== ====================================================
```