



Grandstream Networks, Inc.

---

UCM6xxx IP-PBX Series

**CDR Real-time Output Feature Guide**



## Table of Contents

<b>INTRODUCTION</b> .....	<b>3</b>
<b>IMPLEMENTATION</b> .....	<b>4</b>
Description .....	4
Configuration .....	5
<b>EXAMPLE OF A TEST SCRIPT</b> .....	<b>6</b>

## Table of Figures

Figure 1: CDR Real-time Output Settings .....	5
Figure 2: CDR Recording Output Example .....	6

## Table of Tables

Table 1: CDR Real-time Output Settings .....	5
--	---



## INTRODUCTION

This document introduces the CDR Real-Time Output feature added starting from firmware 1.0.17.16 on UCM62xx/UCM6510 and starting from firmware 1.0.5.4 on UCM630xA. While there are currently no issues with using UCM's existing CDR tools for generating reports, it is highly recommended to eventually implement integration with CDR Real-Time Output, which ensures that no records are overlooked and provides a simpler interface to work with than current methods. Additionally, this may improve 3rd party CDR integration and CTI application functionality.

This new feature uses a real-time CDR output module for Asterisk 11+ and offers the following functionality:

- Connects to the configured server address/port and delivers CDR strings as soon as they are available in Asterisk.
- Has buffer support. If the configured server is not available at the time of delivery, a buffer file will be created and will be delivered as soon as a connection is established.



## IMPLEMENTATION

### Description

To use this new feature, users need to deploy a CDR server to receive the records in real time. The server's IP address and the port number will then need to be configured on the UCM in Value-Added Features→API Configuration→CDR Real-Time Output Settings. When a CDR entry is generated, the UCM will immediately push the generated record to the configured server.

If the server connection fails, records will be cached in the UCM until the connection is reestablished. In order to avoid having the cache become too large, up to 10,000 records can be cached.

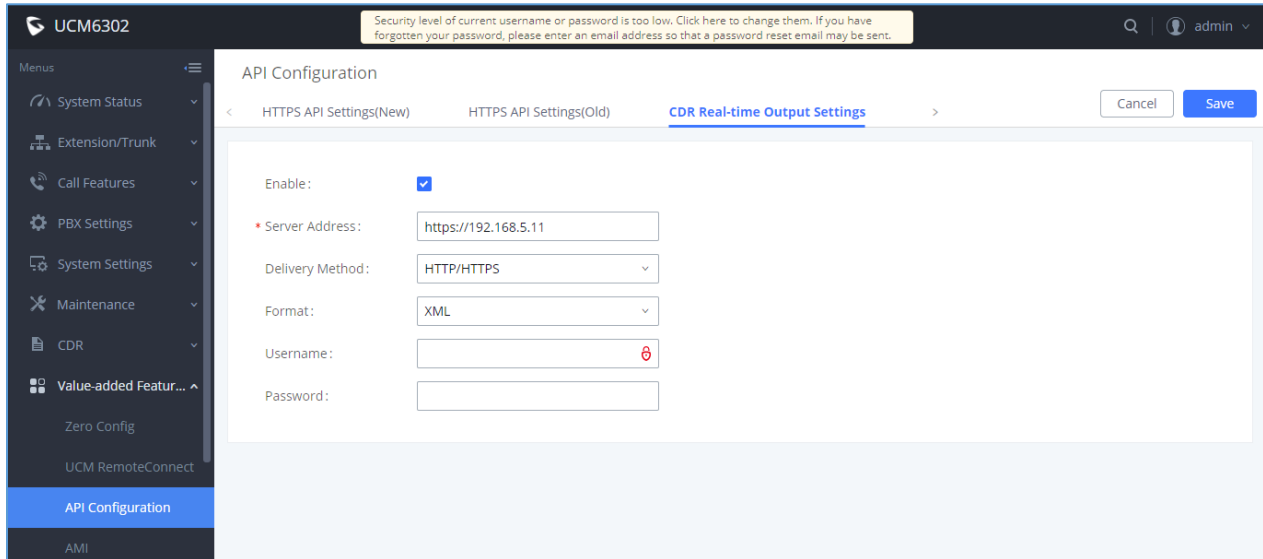
Currently, all available CDR fields and values are sent to the configured server. The following CDR fields will be sent when records are generated:

- accountcode
- src
- dst
- dcontext
- clid
- channel
- dstchannel
- lastapp
- lastdata
- start
- answer
- end
- duration
- billsec
- disposition
- amaflags
- uniqueid
- userfieldchannel\_ext
- dstchannel\_ext
- service
- caller\_name
- recordfiles
- dstanswer
- chanext
- dstchanext
- session
- action\_owner
- action\_type
- src\_trunk\_name
- dst\_trunk\_name



## Configuration

Login to the UCM6XXX's Web interface and navigate to Value-Added Features→API Configuration→CDR Real-Time Output Settings.



The screenshot shows the 'API Configuration' page for 'CDR Real-time Output Settings'. The 'Enable' checkbox is checked. The 'Server Address' is set to 'https://192.168.5.11'. The 'Delivery Method' is set to 'HTTP/HTTPS'. The 'Format' is set to 'XML'. The 'Username' and 'Password' fields are empty. There are 'Cancel' and 'Save' buttons at the top right.

**Figure 1: CDR Real-time Output Settings**

On this page users can configure the CDR receiving server details so the UCM can send CDR details in Real Time to the configured Server. users can either send the data via TCP or HTTP/HTTPS in JSON and XML formats, below are fields and options description.

**Table 1: CDR Real-time Output Settings**

CDR Real-time Output Settings	
<b>Enable</b>	Enables real-time CDR output module. This module connects to selected IP addresses and ports and posts CDR strings as soon as it is available.
<b>Server Address</b>	CDR server IP address
<b>Port</b>	CDR server IP port
<b>Delivery Method</b>	Choose either TCP protocol or HTTP/HTTPS protocol
<b>Format</b>	Choose either XML or JSON for CDR format
<b>Username</b>	Enter the Username for authentication.
<b>Password</b>	Enter the Password for authentication.



## EXAMPLE OF A TEST SCRIPT

The following is a simple Python script to simulate the receiving CDR server.

```
#!/bin/python

# To test the cdr realtime sending function

import string import urllib2 import time import socket

port = raw_input("Please enter the bind port like: 1000\n")

#wait connect

def test():

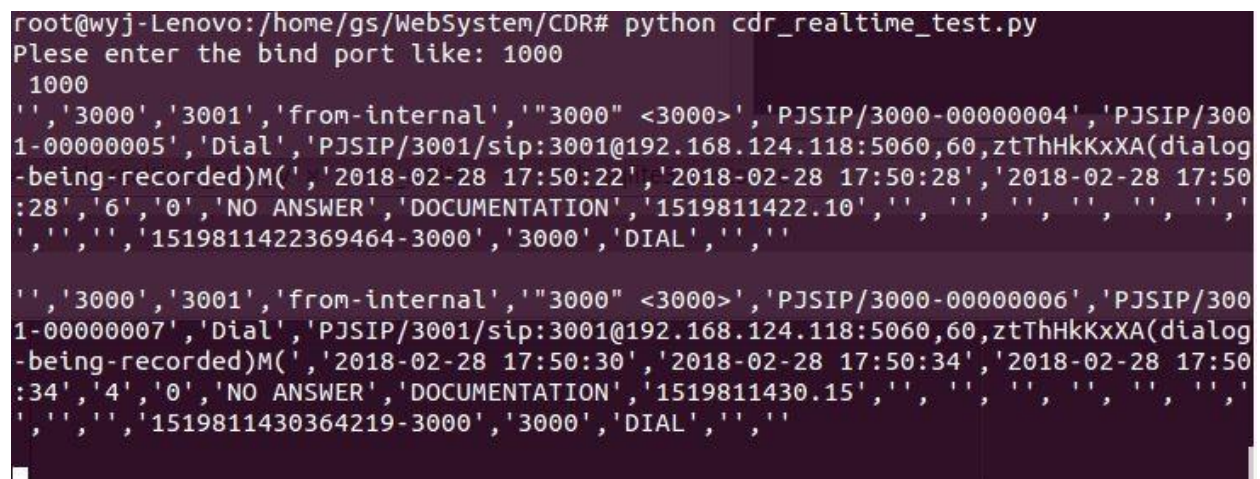
server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
address = ('192.168.124.16', int(port)) //the local IP address
server.bind(address)
server.listen(10)

while 1:

s, addr=server.accept()
data = s.recv(2048) print (data)
s.close()

test()
```

Whenever there is a call ending, the server will receive the corresponding CDR recording:



```
root@wyj-Lenovo:/home/gs/WebSystem/CDR# python cdr_realtime_test.py
Plese enter the bind port like: 1000
1000
', '3000', '3001', 'from-internal', '"3000" <3000>', 'PJSIP/3000-00000004', 'PJSIP/300
1-00000005', 'Dial', 'PJSIP/3001/sip:3001@192.168.124.118:5060,60,ztThHkKxXA(dialog
-being-recorded)M(', '2018-02-28 17:50:22', '2018-02-28 17:50:28', '2018-02-28 17:50
:28', '6', '0', 'NO ANSWER', 'DOCUMENTATION', '1519811422.10', '', '', '', '', '', ''
', '', '', '1519811422369464-3000', '3000', 'DIAL', '', ''

', '3000', '3001', 'from-internal', '"3000" <3000>', 'PJSIP/3000-00000006', 'PJSIP/300
1-00000007', 'Dial', 'PJSIP/3001/sip:3001@192.168.124.118:5060,60,ztThHkKxXA(dialog
-being-recorded)M(', '2018-02-28 17:50:30', '2018-02-28 17:50:34', '2018-02-28 17:50
:34', '4', '0', 'NO ANSWER', 'DOCUMENTATION', '1519811430.15', '', '', '', '', '', ''
', '', '', '1519811430364219-3000', '3000', 'DIAL', '', ''
```

Figure 2: CDR Recording Output Example

