TransNexus

OSP Toolkit

Device Enrollment

Release 2.5.5

09 February 2002

OSP Toolkit

Device Enrollment

Release 2.5.5

09 February 2002

Document 0300-1241-0200

TransNexus
1140 Hammond Drive, Building E
Suite 5250
Atlanta, GA 30328
USA

Phone: +1 770 671 1888
Fax: +1 770 671 1188

E-mail: support@transnexus.com

Contents

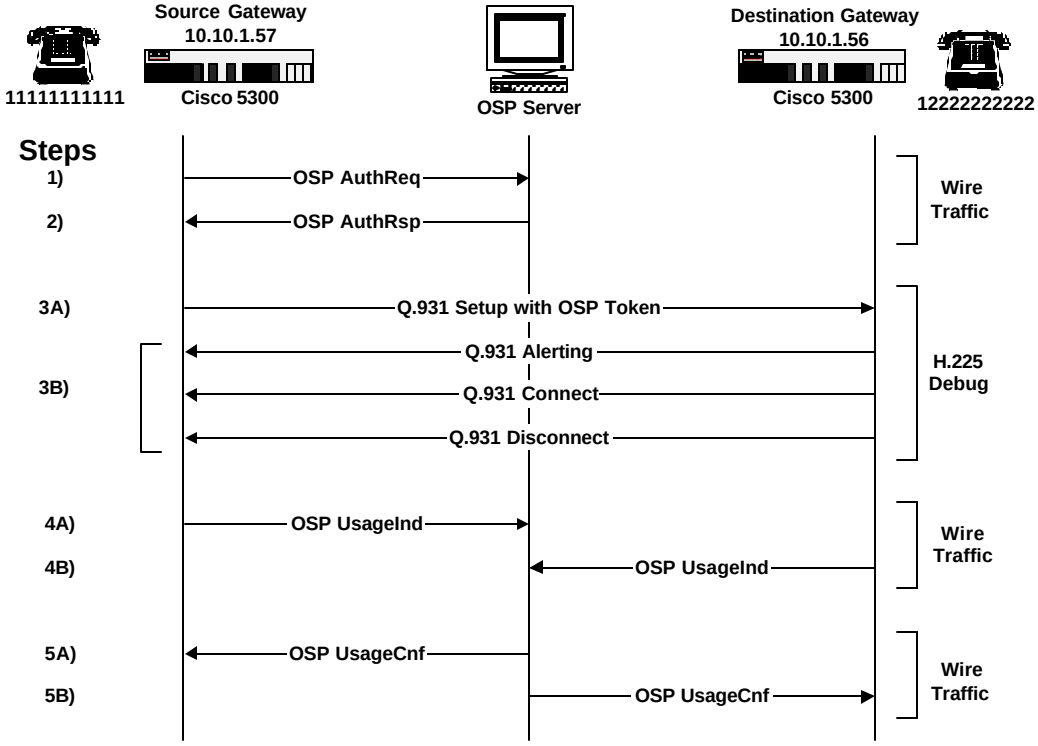
Introduction

The purpose of this document is to provide an example of how the OSP standard has been implemented in generally available products. Cisco's OSP implementation has been selected for this example. Cisco Systems is one of the co-developers of the OSP standard and the leading provider of VoIP gateways worldwide.

The document begins with a description of the testbed network used to create the example call scenario. The second section describes the messages exchanged among the OSP Server and gateways for the example call scenario. The final section provides information on how the OSP token must be formatted in call set-up message for interoperability with Cisco devices.

Test Network and Example Call Scenario

The following diagram illustrates the test network used to create the example call scenario. The VoIP devices used are two Cisco AS5300 H.323 gateways. The source gateway has address 10.10.1.57, and the destination gateway has address 10.10.1.56. Both gateways are running IOS version 12.2(3), and have been enrolled with an OSP Nexus Server. The call scenario begins with a call made from the telephone connected to the source gateway; The calling number is 11111111111, and the called number is 12222222222.



Step 1: Source gateway sends OSP AuthorizationRequest to OSP Server requesting IP address of gateways that can complete the call to the called number 1222222222.

Step 2: OSP Server sends OSP AuthorizationResponse to source gateway with IP address of the destination gateway and an authorization token.

Step 3: Source gateway sends Q.931 call set-up message to destination gateway. Call is completed and disconnected after 21 seconds.

Step 4: Both the source and destination gateways send OSP UsageIndication messages, reporting call duration, to the OSP server.

Step 5: OSP Server sends OSP UsageConfirmation to both source and destination gateways.

Detailed Message Examples

This section provides detailed examples of the five steps in the H.323 call scenario described above. Steps 1, 2, 4 and 5 are OSP messages captured off the wire using a protocol analyzer. Steps 3A and 3B are examples of the Q.931 messages exchanged between the source and destination gateways. These messages were captured from H.225 debug out from the Cisco gateways.

Step 1. OSP AuthorizationRequest

From source gateway 10.10.1.57 to OSP Nexus Server.

```
<Message messageId="23711750491" random="4747">
  <AuthorisationRequest componentId="2371175042768">
    <Timestamp>2001-10-03T22:05:12Z</Timestamp>
    <CallId encoding="base64">jPm35LeBEdWALdS27YGXCQ==</CallId>
    <SourceInfo type="e164">11111111111</SourceInfo>
    <DestinationInfo type="e164">12222222222</DestinationInfo>
    <Service/>
    <MaximumDestinations>3</MaximumDestinations>
  </AuthorisationRequest>
</Message>
```

Step 2. OSP AuthorizationResponse

From the OSP Nexus Server to source gateway 10.10.1.57.

```
<Message messageId='23711750491' random='8946'>
  <AuthorisationResponse componentId='2371175042768'>
    <Timestamp>2001-10-03T22:05:12Z</Timestamp>
    <Status>
      <Description>SUCCESS</Description>
      <Code>200</Code>
    </Status>
    <TransactionId>4304187353840953091</TransactionId>
    <Destination>
      <CallId encoding='base64'>jPm35LeBEdWALdS27YGXCQ==</CallId>
      <DestinationInfo type='transport'>12222222222</DestinationInfo>
      <DestinationSignalAddress>[10.10.1.56]</DestinationSignalAddress>
      <Token encoding='base64'>
MIID4QYJKoZIhvcNAQcCoIID0jCCA84CP\310eAQExDjAMBggqhkiG9w0CBQUAMIIB5QYJK
oZIhvcNAQcBoIIB1gSCAdI8P3htbCB2ZXJzaW9uPScxLjAnPz48VG9rZW5JbmZvIHJhbmRv
```

```
bT0nODk0Nic+PFNvdXJjZUluZm8gdHlwZT0nZTE2NCc+MTExMTExMTExMTE8L1NvdXJjZUl
uZm8+PERlc3RpbmF0aW9uSW5mbyB0eXBlPSd0cmFuc3BvcnQnPjEyMjIyMjIyMjIyPC9EZX
N0aW5hdGlvbkluZm8+PENhbGxJZCBlbmNvZGluZz0nYmFzZTY0Jz5qUG0zNUxlQkVkV0FMZ
FMyN1lHWENRPT08L0NhbGxJZD48VmFsaWRBZnRlcj4yMDAxLTEwLTAzVDIyOjAwOjEyWjwv
VmFsaWRBZnRlcj48VmFsaWRVbnRpbD4yMDAxLTEwLTAzVDIyOjEwOjEyWjwvVmFsaWRVbnR
pbD48VHJhbnNhY3Rpb25JZD40MzA0MTg3MzUzODQwOTUzMDkxPC9UcmFuc2Fj09dGlvbklk
PjxVc2FnZURldGFpbD48QW1vdW50PjE0NDAwPC9BbW91bnQ+PEluY3JlbWVudD4xPC9JbmN
yZW1lbnQ+PFNlcnZpY2UvPjxVbml0PnM8L1VuaXQ+PC9Vc2FnZURldGFpbD48L1Rva2VuSW
5mbz4AoIIBOTCCATUwgeACAQEwDQYJKoZIhvcNAQEEBQAwJjEQMA4GA1UEAxMHYmV0YWJlM
TESMBAGA1UEChMJT1NQU2VydmVyMB4XDTAxMDkyMDExMTIwNloXDTAzMDkyMTExMTIwNlow
JjEQMA4GA1UEAxMHYmV0YWJlMTESMBAGA1UEChMJT1NQU2VydmVyMFwwDQYJKoZIhvcNAQE
BBQADSwAwSAJBAPGeGwV41EIhX0jEDFLRXQhDEr50OUQPq+f55VwQd0TQNts06BP29+UiNd
RW3c3IRHdZcJdC1Cg68ME9cgeq0h8CAwEAATANBgkqhkiG9w0BAQQFAANBAGb8Raf/AKuMq
CMD0rr2UnGjNBOaefsg6C/w7qp0FFombGdAK19W6tTS7xqCj1t/+QeSFSH9r+W03GmO7H18
iQ9PIxgZIwgY8CAQEwKzAmMRAwDgYDVQQDEwdiZXRhYmUxMRIwEAYDVQQKEwlPU1BTZXJ2Z
XICAQEwDAYIKoZIhvcNAgUFADANBgkqhkiG9w0BAQEFAARATHbkgCDrEJGGNHwl/cBcOBib
I+kAktz5HgNP8s5PrfCtFBh1PjqTxPt4tEcPCFEtF9I9cd0dURuCM14LItQwXw==
```


```
      </Token>
      <UsageDetail>
        <Amount>14400</Amount>
        <Increment>1</Increment>
        <Service/>
        <Unit>s</Unit>
      </UsageDetail>
      <ValidAfter>2001-10-03T22:00:12Z</ValidAfter>
      <ValidUntil>2001-10-03T22:10:12Z</ValidUntil>
    </Destination>
  </AuthorisationResponse>
</Message>
```

Step 3A. Q.931 Call Set-up with token

The messages below are H.225 debug output from source gateway 10.10.1.57. The OSP token returned to the source gateway in the OSP AuthorizationRequest from the OSP server is packaged in the Q.931 call set-up message from the source gateway to the destination gateway.

H.225 Event Messages debugging is on
H.225 ASN1 Messages debugging is on

```
GATEWAY1#Changing to new event: CONNECT
h323chan_chn_connect: connecting to 10.10.1.56:1720

Oct  3 22:05:12.401:       h323chan_gw_conn: connect in progress on
socket [1]h323chan_chn_connect: using fd 1, owner_data(ccb) 0x624BA0E8
changing from NONE state to CONNECTING state
h323chan_chn_process_read_socket: fd (1) of type CONNECT_PENDING has
data
Changing to new event: CONNECTED
changing from CONNECTING state to CONNECTED state

Oct  3 22:05:12.405: compose_new_style_settlement_token: Building
standard settlement token.
```

```
Oct  3 22:05:12.405: H225.0 OUTGOING PDU ::=

value H323_UserInformation ::=
    {
      h323-uu-pdu
      {
        h323-message-body setup :
        {
          protocolIdentifier { 0 0 8 2250 0 2 }
          sourceInfo
          {
            gateway
            {
              protocol
              {
                voice :
                {
                  supportedPrefixes
                  {
                  }
                }
              }
            }
            mc FALSE
            undefinedNode FALSE
          }
          activeMC FALSE
          conferenceID '8CF9B7E4B78111D5802CD4B6ED819709'H
          conferenceGoal create : NULL
          callType pointToPoint : NULL
          sourceCallSignalAddress ipAddress :
          {
            ip '0A0A0139'H
            port 11011
          }
          callIdentifier
          {
            guid '8CF9B7E4B78111D5802DD4B6ED819709'H
          }
          tokens
          {

            {
              tokenOID { 0 4 0 1321 1 2 }
              nonStandard
              {
                nonStandardIdentifier { 0 4 0 1321 1 2 }
                data '308203E106092A864886F70D010702A08203D230...'H
              }
            }
          }
          fastStart
          {
            '0000000D4001800A040001000A0A013946E9'H,
            '400000060401004D4001801114000l000A0A0139...'H
```

```
          }
          mediaWaitForConnect FALSE
          canOverlapSend FALSE
        }
        h245Tunneling FALSE
      }
    }
```

```
Oct  3 22:05:12.413: H225.0 OUTGOING ENCODE BUFFER::= 20 80060008
914A0002 0880013C 05010000 8CF9B7E4 B78111D5 802CD4B6 ED819709 00455C07
000A0A01 392B0311 008CF9B7 E4B78111 D5802DD4 B6ED8197 0983F801 00800604
008A2901 02060400 8A290102 83E53082 03E10609 2A864886 F70D0107 02A08203
D2308203 CE020101 310E300C 06082A86 4886F70D 02050500 308201E5 06092A86
4886F70D 010701A0 8201D604 8201D23C 3F786D6C 20766572 73696F6E 3D27312E
30273F3E 3C546F6B 656E496E 666F2072 616E646F 6D3D2738 39343627 3E3C536F
75726365 496E666F 20747970 653D2765 31363427 3E313131 31313131 31313131
3C2F536F 75726365 496E666F 3E3C4465 7374696E 6174696F 6E496E66 6F207479
70653D27 7472616E 73706F72 74273E31 32323232 32323232 32323C2F 44657374
696E6174 696F6E49 6E666F3E 3C43616C 6C496420 656E636F 64696E67 3D276261
73653634 273E6A50 6D33354C 65424564 57414C64 53323759 47584351 3D3D3C2F
43616C6C 49643E3C 56616C69 64416674 65723E32 3030312D 31302D30 33543232
3A30303A 31325A3C 2F56616C 69644166 7465723E 3C56616C 6964556E 74696C3E
32303031 2D31302D 30335432 323A3130 3A31325A 3C2F5661 6C696455 6E74696C
3E3C5472 616E7361 6374696F 6E49643E 34333034 31383733 35333834 30393533
3039313C 2F547261 6E736163 74696F6E 49643E3C 55736167 65446574 61696C3E
3C416D6F 756E743E 31343430 303C2F41 6D6F756E 743E3C49 6E637265 6D656E74
3E313C2F 496E6372 656D656E 743E3C53 65727669 63652F3E 3C556E69 743E733C
2F556E69 743E3C2F 55736167 65446574 61696C3E 3C2F546F 6B656E49 6E666F3E
00A08201 39308201 353081E0 02010130 0D06092A 864886F7 0D010104 05003026
3110300E 06035504 03130762 65746162 65313112 30100603 55040A13 094F5350
53657276 6572301E 170D3031 30393230 31313132 30365A17 0D303330 39323131
31313230 365A3026 3110300E 06035504 03130762 65746162 65313112 30100603
55040A13 094F5350 53657276 6572305C 300D0609 2A864886 F70D0101 01050003
4B003048 024100F1 9E1B0578 D442215F 48C40C52 D15D0843 12BE7439 440FABE7
F9E55C10 7744D036 DB34E813 F6F7E522 35D456DD CDC84477 59709742 D4283AF0
C13D7207 AAD21F02 03010001 300D0609 2A864886 F70D0101 04050003 410066FC
45A7FF00 AB8CA823 03D2BAF6 5271A334 139AB20E 82FF0EEA A74145A2 66C67402
B5F56EAD 4D2EF1A8 28F5B7FF 90792152 1FDAFE5B 4DC698EE C7D7C890 F4F23181
9230818F 02010130 2B302631 10300E06 03550403 13076265 74616265 31311230
10060355 040A1309 4F535053 65727665 72020101 300C0608 2A864886 F70D0205
0500300D 06092A86 4886F70D 01010105 0004404C 76E48020 EB109186 347C25FD
C05C3818 9B23E900 92DCF91E 034FF2CE 4FADF0AD 1418753E 3A93C4FB 78B4470F
08512D17 D23D71DD 1D511B82 335E0B22 D4305F32 02120000 000D4001 800A0400
01000A0A 013946E9 1D400000 06040100 4D400180 11140001 000A0A01 3946E800
0A0A0139 46E90100 01000680 0100
Oct  3 22:05:12.437:
Hex representation of the SETUP TPKT to
send.030004A20802000B0504038090A36C0D2180313131313131313131313131700CA131
323232323232323232327E0474052080060008914A00020880013C050100008CF9B7E4B
78111D5802CD4B6ED81970900455C07000A0A01392B0311008CF9B7E4B78111D5802DD4
B6ED81970983F80100800604008A2901020604008A29010283E5308203E106092A86488
6F70D010702A08203D2308203CE020101310E300C06082A864886F70D02050500308201
E506092A864886F70D010701A08201D6048201D23C3F786D6C2076657273696F6E3D273
```

```
12E30273F3E3C546F6B656E496E666F2072616E646F6D3D2738393436273E3C536F7572
6365496E666F20747970653D2765313634273E313131313131313131313C2F536F757
26365496E666F3E3C44657374696E6174696F6E496E666F20747970653D277472616E73
706F7274273E313232323232323232323C2F44657374696E6174696F6E496E666F3E3
C43616C6C496420656E636F64696E673D27626173653634273E6A506D33354C65424564
57414C64533237594758435 13D3D3C2F43616C6C49643E3C56616C696441667465723E3
23030312D31302D30335432323A30303A31325A3C2F56616C696441667465723E3C5661
6C6964556E74696C3E323030312D31302D30335432323A31303A31325A3C2F56616C696
4556E74696C3E3C5472616E73616374696F6E49643E343330343138373353338343039
35333039313C2F5472616E73616374696F6E49643E3C5573616765446574616C3E3C4
16D6F756E743E31343430303C2F416D6F756E743E3C496E6372656D656E743E313C2F49
6E6372656D656E743E536572766963652F3E3C556E69743E733C2F556E69743E3C2F5
5736167654465746 1696C3E3C2F546F6B656E496E666F3E00A0820139308201353081E0
020101300D06092A864886F70D0101040500302631103 00E060355040313076265746 16
2653131123010060355040A13094F535053657276657230 1E170D303130393230313131
3230365A170D303330393231313131323036 5A30263110300E060355040313076265746
162653131123010060355040A13094F535053657276657230 5C300D06092A864886F70D
0101010500034B003048024100F19E1B0578D442215F48C40C52D15D084312BE7439440
FABE7F9E55C107744D036DB34E813F6F7E52235D456DDCDC8447759709742D4283AF0C1
3D7207AAD21F02030100013 00D06092A864886F70D0101040500034 10066FC45A7FF00A
B8CA82303D2BAF65271A334139AB20E82FF0EEAA74145A266C67402B5F56EAD4D2EF1A8
28F5B7FF907921521FDAFE5B4DC698EEC7D7C890F4F231819230818F020101302B30263
110300E060355040313076265746162653131123010060355040A13094F535053657276
6572020101300C06082A864886F70D02050500300D06092A864886F70D0101010500044
04C76E48020EB109186347C25FDC05C38189B23E90092DCF91E034FF2CE4FADF0AD1418
753E3A93C4FB78B4470F08512D17D23D71DD1D511B82335E0B22D4305F3202120000000
D4001800A040001000A0A013946E91D400000060401004D4001801114000 1000A0A0139
46E8000A0A013946E90100010006800100
Oct  3 22:05:12.457: h225SetupRequest: Q.931 SETUP sent from socket
[1]h323chan_chn_process_read_socket: fd (1) of type CONNECTED has data

Oct  3 22:05:12.457: H323chan Lib: No Data on socket [1]:
PROCESS_READ: NOT COMPLETE, rc 10, fd=1
h323chan_chn_process_read_socket: fd (1) of type CONNECTED has data

Hex representation of the received
TPKT0300006908028 00B027E005D052180060008914A000200048811008CF9B7E4B7811
1D5802DD4B6ED819709390219 0000000D4001801114000 1000A0A013843D8000A0A0138
43D91D400000060401004D4001801114000 1000A0A013946E8000A0A013843D90680010
0
Oct  3 22:05:12.557: h225ParseData: Q.931 CALL PROCEEDING received on
socket [1]
Oct  3 22:05:12.561: H225.0 INCOMING ENCODE BUFFER::= 21 80060008
914A0002 00048811 008CF9B7 E4B78111 D5802DD4 B6ED8197 09390219 0000000D
40018011 14000100 0A0A0138 43D8000A 0A013843 D91D4000 00060401 004D4001
80111400 01000A0A 013946E8 000A0A01 3843D906 800100
Oct  3 22:05:12.561:
Oct  3 22:05:12.561: H225.0 INCOMING PDU ::=

value H323_UserInformation ::=
    {
      h323-uu-pdu
      {
        h323-message-body callProceeding :
        {
```

```
            protocolIdentifier { 0 0 8 2250 0 2 }
            destinationInfo
            {
              mc FALSE
              undefinedNode FALSE
            }
            callIdentifier
            {
              guid '8CF9B7E4B78111D5802DD4B6ED819709'H
            }
            fastStart
            {
              '0000000D40018011140001000A0A013843D8000A...'H,
              '400000060401004D40018011140001000A0A0139...'H
            }
          }
          h245Tunneling FALSE
        }
      }


h323chan_chn_process_read_socket: fd (1) of type CONNECTED has data

Hex representation of the received
TPKT03000320802800B013401017E00230523800600089l4A000200048011008CF9B7E
4B78111D5802DD4B6ED81970906800100
Oct  3 22:05:12.973: h225ParseData: Q.931 ALERTING received on socket
[1]
Oct  3 22:05:12.973: H225.0 INCOMING ENCODE BUFFER::= 23 80060008
914A0002 00048011 008CF9B7 E4B78111 D5802DD4 B6ED8197 09068001 00
Oct  3 22:05:12.977:
Oct  3 22:05:12.977: H225.0 INCOMING PDU ::=

value H323_UserInformation ::=
      {
        h323-uu-pdu
        {
          h323-message-body alerting :
          {
            protocolIdentifier { 0 0 8 2250 0 2 }
            destinationInfo
            {
              mc FALSE
              undefinedNode FALSE
            }
            callIdentifier
            {
              guid '8CF9B7E4B78111D5802DD4B6ED819709'H
            }
          }
          h245Tunneling FALSE
        }
      }
```

```
h323chan_chn_process_read_socket: fd (1) of type CONNECTED has data

Hex representation of the received
TPKT0300004B0802800B0704038090A37E003A052280060008914A00020880013C05010
0008CF9B7E4B78111D5802CD4B6ED819709090011008CF9B7E4B78111D5802DD4B6ED81
970906800100
Oct  3 22:05:13.173: h225ParseData: Q.931 CONNECT received on socket
[1]
Oct  3 22:05:13.173: H225.0 INCOMING ENCODE BUFFER::= 22 80060008
914A0002 0880013C 05010000 8CF9B7E4 B78111D5 802CD4B6 ED819709 09001100
8CF9B7E4 B78111D5 802DD4B6 ED819709 06800100
Oct  3 22:05:13.177:
Oct  3 22:05:13.177: H225.0 INCOMING PDU ::=

value H323_UserInformation ::=
    {
      h323-uu-pdu
      {
        h323-message-body connect :
        {
          protocolIdentifier { 0 0 8 2250 0 2 }
          destinationInfo
          {
            gateway
            {
              protocol
              {
                voice :
                {
                  supportedPrefixes
                  {
                  }
                }
              }
            }
            mc FALSE
            undefinedNode FALSE
          }
          conferenceID '8CF9B7E4B78111D5802CD4B6ED819709'H
          callIdentifier
          {
            guid '8CF9B7E4B78111D5802DD4B6ED819709'H
          }
        }
        h245Tunneling FALSE
      }
    }



Oct  3 22:05:13.201: %ISDN-6-CONNECT: Interface Serial0:0 is now
connected to 11111111111
Oct  3 22:05:19.201: %ISDN-6-CONNECT: Interface Serial0:0 is now
connected to 11111111111
Oct  3 22:05:34.722: %ISDN-6-DISCONNECT: Interface Serial0:0
```

```
disconnected from 11111111111 , call lasted 21 seconds
Oct  3 22:05:34.726: H225.0 OUTGOING PDU ::=

value H323_UserInformation ::=
    {
      h323-uu-pdu
      {
        h323-message-body releaseComplete :
        {
          protocolIdentifier { 0 0 8 2250 0 2 }
          callIdentifier
          {
            guid '8CF9B7E4B78111D5802DD4B6ED819709'H
          }
        }
        h245Tunneling FALSE
      }
    }



Oct  3 22:05:34.730: H225.0 OUTGOING ENCODE BUFFER::= 25 80060008
914A0002 0111008C F9B7E4B7 8111D580 2DD4B6ED 81970906 800100
Oct  3 22:05:34.730:
Hex representation of the RELEASE COMPLETE TPKT to
send.030000310802000B5A080280907E00210525800600008914A00020111008CF9B7E4
B78111D5802DD4B6ED81970906800100
Oct  3 22:05:34.730: h225TerminateRequest: Q.931 RELEASE COMPLETE sent
from socket [1]. Call state changed to [Null].
Oct  3 22:05:34.730:       h323chan_close: TCP connection from socket
[1] closed
```

Step 3B. Q.931 messages from destination gateway

Cisco H.225 debug output from destination gateway 10.10.1.56.
H.225 Event Messages debugging is on
H.225 ASN1 Messages debugging is on

```
GATEWAY2#h323chan_chn_process_read_socket: fd (0) of type LISTENING has
data
Changing to new event: ACCEPT
h323chan_chn_accept: 0

Oct  3 22:05:12.397:       h323chan_gw_accept: TCP connection accepted
from 10.10.1.57:11011 on socket [1]
Oct  3 22:05:12.397: local(0x0) accepts TCP conn from
10.10.1.57(0xA0A0139) port 11011changing from LISTENING state to
ACCEPTED state
h323chan_chn_process_read_socket: fd (1) of type ACCEPTED has data

Hex representation of the received
TPKT030004A20802000B0504038090A36C0D2180313131313131313131313131700CA1313
232323232323232327E0474052080060008914A00020880013C050100008CF9B7E4B7
```

```
8111D5802CD4B6ED81970900455C07000A0A01392B0311008CF9B7E4B78111D5802DD4B
6ED81970983F80100800604008A2901020604008A29010283E5308203E106092A864886
F70D010702A08203D2308203CE020101310E300C06082A864886F70D02050500308201E
506092A864886F70D010701A08201D6048201D23C3F786D6C2076657273696F6E3D2731
2E30273F3E3C546F6B656E496E666F2072616E646F6D3D2738393436273E3C536F75726
365496E666F20747970653D2765313634273E3131313131313131313131313C2F536F7572
6365496E666F3E3C44657374696E6174696F6E496E666F20747970653D277472616E737
06F7274273E31323232323232323232323C2F44657374696E6174696F6E496E666F3E3C
43616C6C496420656E636F64696E673D27626173653634273E6A506D33354C654245645
7414C64533237594758435 13D3D3C2F43616C6C49643E3C56616C696441667465723E32
3030312D31302D30335432323A30303A31325A3C2F56616C696441667465723E3C56616
C6964556E74696C3E323030312D31302D30335432323A31303A31325A3C2F56616C6964
556E74696C3E3C5472616E73616374696F6E49643E343330343138373333533383430393
5333039313C2F5472616E73616374696F6E49643E3C5573616765446574657461696C3E3C41
6D6F756E743E31343430303C2F416D6F756E743E3C496E6372656D656E743E313C2F496
E6372656D656E743E536572766963652F3E3C556E69743E733C2F556E69743E3C2F55
736167654465746 1696C3E3C2F546F6B656E496E666F3E00A0820139308201353081E00
20101300D06092A864886F70D0101040500302631 10300E06035504031307626574616265313112301006035504 0313076265746162
65313112301006035504 0A13094F53505365727665723 01E170D3031303932303131313
230365A170D3033303932313131313230365A30263110300E06035504031307626574616
62653131123010060355040A13094F53505365727665723 05C300D06092A864886F70D0
101010500034B003048024100F19E1B0578D442215F48C40C52D15D084312BE7439440F
ABE7F9E55C107744D036DB34E813F6F7E52235D456DDCDC8447759709742D4283AF0C13
D7207AAD21F02030100013 00D06092A864886F70D0101040500034 10066FC45A7FF00AB
8CA82303D2BAF65271A334139AB20E82FF0EEAA74145A266C67402B5F56EAD4D2EF1A82
8F5B7FF907921521FDAFE5B4DC698EEC7D7C890F4F231819230818F020101302B302631
10300E06035504031307626574616265313112301006035504 0A13094F5350536572766
572020101300C06082A864886F70D02050500300D06092A864886F70D01010105000440
4C76E48020EB109186347C25FDC05C38189B23E90092DCF91E034FF2CE4FADF0AD14187
53E3A93C4FB78B4470F08512D17D23D71DD1D511B82335E0B22D4305F3202120000000D
4001800A040001000A0A013946E91D40000006040 1004D400180111400010 00A0A01394
6E8000A0A013946E90100010006800100
Oct  3 22:05:12.473: h225ParseData: Q.931 SETUP received on socket [1]
Oct  3 22:05:12.473: H225.0 INCOMING ENCODE BUFFER::= 20 80060008
914A0002 0880013C 05010000 8CF9B7E4 B78111D5 802CD4B6 ED819709 00455C07
000A0A01 392B0311 008CF9B7 E4B78111 D5802DD4 B6ED8197 0983F801 00800604
008A2901 02060400 8A290102 83E53082 03E10609 2A864886 F70D0107 02A08203
D2308203 CE020101 310E300C 06082A86 4886F70D 02050500 308201E5 06092A86
4886F70D 010701A0 8201D604 8201D23C 3F786D6C 20766572 73696F6E 3D27312E
30273F3E 3C546F6B 656E496E 666F2072 616E646F 6D3D2738 39343627 3E3C536F
75726365 496E666F 20747970 653D2765 31363427 3E313131 31313131 31313131
3C2F536F 75726365 496E666F 3E3C4465 7374696E 6174696F 6E496E66 6F207479
70653D27 7472616E 73706F72 74273E31 32323232 32323232 32323C2F 44657374
696E6174 696F6E49 6E666F3E 3C43616C 6C496420 656E636F 64696E67 3D276261
73653634 273E6A50 6D33354C 65424564 57414C64 53323759 47584351 3D3D3C2F
43616C6C 49643E3C 56616C69 64416674 65723E32 3030312D 31302D30 33543232
3A30303A 31325A3C 2F56616C 69644166 7465723E 3C56616C 6964556E 74696C3E
32303031 2D31302D 30335432 323A3130 3A31325A 3C2F5661 6C696455 6E74696C
3E3C5472 616E7361 6374696F 6E49643E 34333034 31383733 35333834 30393533
3039313C 2F547261 6E736163 74696F6E 49643E3C 55736167 65446574 61696C3E
3C416D6F 756E743E 31343430 303C2F41 6D6F756E 743E3C49 6E637265 6D656E74
3E313C2F 496E6372 656D656E 743E3C53 65727669 63652F3E 3C556E69 743E733C
2F556E69 743E3C2F 55736167 65446574 61696C3E 3C2F546F 6B656E49 6E666F3E
00A08201 39308201 353081E0 02010130 0D06092A 864886F7 0D010104 05003026
3110300E 06035504 03130762 65746162 65313112 30100603 55040A13 094F5350
```

```
53657276 6572301E 170D3031 30393230 31313132 30365A17 0D303330 39323131
31313230 365A3026 3110300E 06035504 03130762 65746162 65313112 30100603
55040A13 094F5350 53657276 6572305C 300D0609 2A864886 F70D0101 01050003
4B003048 024100F1 9E1B0578 D442215F 48C40C52 D15D0843 12BE7439 440FABE7
F9E55C10 7744D036 DB34E813 F6F7E522 35D456DD CDC84477 59709742 D4283AF0
C13D7207 AAD21F02 03010001 300D0609 2A864886 F70D0101 04050003 410066FC
45A7FF00 AB8CA823 03D2BAF6 5271A334 139AB20E 82FF0EEA A74145A2 66C67402
B5F56EAD 4D2EF1A8 28F5B7FF 90792152 1FDAFE5B 4DC698EE C7D7C890 F4F23181
9230818F 02010130 2B302631 10300E06 03550403 13076265 74616265 31311230
10060355 040A1309 4F535053 65727665 72020101 300C0608 2A864886 F70D0205
0500300D 06092A86 4886F70D 01010105 0004404C 76E48020 EB109186 347C25FD
C05C3818 9B23E900 92DCF91E 034FF2CE 4FADF0AD 1418753E 3A93C4FB 78B4470F
08512D17 D23D71DD 1D511B82 335E0B22 D4305F32 02120000 000D4001 800A0400
01000A0A 013946E9 1D400000 06040100 4D400180 11140001 000A0A01 3946E800
0A0A0139 46E90100 01000680 0100
Oct  3 22:05:12.497:
Oct  3 22:05:12.497: H225.0 INCOMING PDU ::=

value H323_UserInformation ::=
    {
      h323-uu-pdu
      {
        h323-message-body setup :
        {
          protocolIdentifier { 0 0 8 2250 0 2 }
          sourceInfo
          {
            gateway
            {
              protocol
              {
                voice :
                {
                  supportedPrefixes
                  {
                  }
                }
              }
            }
            mc FALSE
            undefinedNode FALSE
          }
          activeMC FALSE
          conferenceID '8CF9B7E4B78111D5802CD4B6ED819709'H
          conferenceGoal create : NULL
          callType pointToPoint : NULL
          sourceCallSignalAddress ipAddress :
          {
            ip '0A0A0139'H
            port 11011
          }
          callIdentifier
          {
            guid '8CF9B7E4B78111D5802DD4B6ED819709'H
          }
```

```
            tokens
            {

              {
                tokenOID { 0 4 0 1321 1 2 }
                nonStandard
                {
                  nonStandardIdentifier { 0 4 0 1321 1 2 }
                  data '308203E106092A864886F70D010702A08203D230...'H
                }
              }
            }
            fastStart
            {
              '0000000D4001800A040001000A0A013946E9'H,
              '400000060401004D4001801114000l000A0A0139...'H
            }
            mediaWaitForConnect FALSE
            canOverlapSend FALSE
          }
          h245Tunneling FALSE
        }
      }


Oct  3 22:05:12.505: parse_ClearTokenNonstd: Decoding settlement clear
token using standard format, len=997
Oct  3 22:05:12.541: H225.0 OUTGOING PDU ::=

value H323_UserInformation ::=
      {
        h323-uu-pdu
        {
          h323-message-body callProceeding :
          {
            protocolIdentifier { 0 0 8 2250 0 2 }
            destinationInfo
            {
              mc FALSE
              undefinedNode FALSE
            }
            callIdentifier
            {
              guid '8CF9B7E4B78111D5802DD4B6ED819709'H
            }
            fastStart
            {
              '0000000D4001801114000l000A0A013843D8000A...'H,
              '400000060401004D4001801114000l000A0A0139...'H
            }
          }
          h245Tunneling FALSE
        }
      }
```

```
Oct  3 22:05:12.545: H225.0 OUTGOING ENCODE BUFFER::= 21 80060008
914A0002 00048811 008CF9B7 E4B78111 D5802DD4 B6ED8197 09390219 0000000D
40018011 14000100 0A0A0138 43D8000A 0A013843 D91D4000 00060401 004D4001
80111400 01000A0A 013946E8 000A0A01 3843D906 800100
Oct  3 22:05:12.549:
Hex representation of the CALL PROCEEDING TPKT to
send.030000690802800B027E005D052180060008914A000200048811008CF9B7E4B781
11D5802DD4B6ED819709390219000000D400180111400010000A0A013843D8000A0A013
843D91D4000000604010004D400180111400010000A0A013946E8000A0A013843D9068001
00
Oct  3 22:05:12.549: h225CallProcRequest: Q.931 CALL PROCEEDING sent
fromsocket [1].
Oct  3 22:05:12.961: %ISDN-6-CONNECT: Interface Serial0:22 is now
connected to 12222222222
Oct  3 22:05:12.961: H225.0 OUTGOING PDU ::=

value H323_UserInformation ::=
    {
      h323-uu-pdu
      {
        h323-message-body alerting :
        {
          protocolIdentifier { 0 0 8 2250 0 2 }
          destinationInfo
          {
            mc FALSE
            undefinedNode FALSE
          }
          callIdentifier
          {
            guid '8CF9B7E4B78111D5802DD4B6ED819709'H
          }
        }
        h245Tunneling FALSE
      }
    }



Oct  3 22:05:12.965: H225.0 OUTGOING ENCODE BUFFER::= 23 80060008
914A0002 00048011 008CF9B7 E4B78111 D5802DD4 B6ED8197 09068001 00
Oct  3 22:05:12.965:
Hex representation of the ALERTING TPKT to
send.030000320802800B013401017E00230523800600008914A000200048011008CF9B7
E4B78111D5802DD4B6ED81970906800100
Oct  3 22:05:12.965: h225AlertRequest: Q.931 ALERTING sent from socket
[1]. Call state changed to [Call Received].
Oct  3 22:05:12.973: H225.0 OUTGOING PDU ::=

value H323_UserInformation ::=
    {
      h323-uu-pdu
```

```
        {
          h323-message-body connect :
          {
            protocolIdentifier { 0 0 8 2250 0 2 }
            destinationInfo
            {
              gateway
              {
                protocol
                {
                  voice :
                  {
                    supportedPrefixes
                    {
                    }
                  }
                }
              }
              mc FALSE
              undefinedNode FALSE
            }
            conferenceID '8CF9B7E4B78111D5802CD4B6ED819709'H
            callIdentifier
            {
              guid '8CF9B7E4B78111D5802DD4B6ED819709'H
            }
          }
          h245Tunneling FALSE
        }
      }
```

```
Oct  3 22:05:12.977: H225.0 OUTGOING ENCODE BUFFER::= 22 80060008
914A0002 0880013C 05010000 8CF9B7E4 B78111D5 802CD4B6 ED819709 09001100
8CF9B7E4 B78111D5 802DD4B6 ED819709 06800100
Oct  3 22:05:12.977:
Hex representation of the CONNECT TPKT to
send.0300004B0802800B0704038090A37E003A052280060008914A00020880013C0501
00008CF9B7E4B78111D5802CD4B6ED819709090011008CF9B7E4B78111D5802DD4B6ED8
1970906800100
Oct  3 22:05:12.981: h225SetupResponse: Q.931 CONNECT sent from socket
[1]
Oct  3 22:05:18.961: %ISDN-6-CONNECT: Interface Serial0:22 is now
connected to 12222222222 h323chan_chn_process_read_socket: fd (1) of
type ACCEPTED has data

Hex representation of the received
TPKT0300003108020000B5A080280907E00210525800600008914A00020111008CF9B7E4B
78111D5802DD4B6ED81970906800100
Oct  3 22:05:34.725: h225ParseData: Q.931 RELEASE COMPLETE received on
socket [1]
Oct  3 22:05:34.729: H225.0 INCOMING ENCODE BUFFER::= 25 80060008
914A0002 0111008C F9B7E4B7 8111D580 2DD4B6ED 81970906 800100
Oct  3 22:05:34.729:
```

```
Oct  3 22:05:34.729: H225.0 INCOMING PDU ::=

value H323_UserInformation ::=
    {
      h323-uu-pdu
      {
        h323-message-body releaseComplete :
        {
          protocolIdentifier { 0 0 8 2250 0 2 }
          callIdentifier
          {
            guid '8CF9B7E4B78111D5802DD4B6ED819709'H
          }
        }
        h245Tunneling FALSE
      }
    }


h323chan_chn_process_read_socket: fd (1) of type ACCEPTED has data

Oct  3 22:05:34.733:       h323chan_recvdata:Connection lost socket [1]
Oct  3 22:05:34.733:       h323chan_close: TCP connection from socket
[1] closedh225TerminateRequest: Unidentifiable socket -1
Oct  3 22:05:34.753: %ISDN-6-DISCONNECT: Interface Serial0:22
disconnected from 12222222222 , call lasted 21 seconds
```

Step 4A. OSP UsageIndication from Source

From source gateway, 10.10.1.57, to OSP Nexus Server.

```
<Message messageId="2597352024114" random="4809">
  <UsageIndication componentId="2597352025057">
    <Timestamp>2001-10-03T22:05:34Z</Timestamp>
    <Role>source</Role>
    <TransactionId>4304187353840953091</TransactionId>
    <CallId encoding="base64">jPm35LeBEdWALdS27YGXCQ==</CallId>
    <SourceInfo type="e164">11111111111</SourceInfo>
 <SourceAlternate type="transport">[10.10.1.57]</SourceAlternate>
    <DestinationInfo type="e164">12222222222</DestinationInfo>
 <DestinationAlternate type="transport">[10.10.1.56]</DestinationAlternate>
    <UsageDetail>
      <Service/>
      <Amount>21</Amount>
      <Increment>1</Increment>
      <Unit>s</Unit>
    </UsageDetail>
 <gric.com:TransactionStartTime critical="False">2001-10-03T22:05:13Z
                                              </gric.com:TransactionStartTime>
    <gric.com:TransactionStatus critical="False">
      <gric.com:TransactionCode critical="False">1016
                                            </gric.com:TransactionCode>
   <gric.com:Description critical="False">normal call clearing
                                              </gric.com:Description>
```

```
        </gric.com:TransactionStatus>
        <transnexus.com:Statistics critical="False">
          <transnexus.com:LossSent critical="False">
         <transnexus.com:Packets critical="False">0</transnexus.com:Packets>
         <transnexus.com:Fraction critical="False">0</transnexus.com:Fraction>
       </transnexus.com:LossSent
          <transnexus.com:LossReceived critical="False">
         <transnexus.com:Packets critical="False">0</transnexus.com:Packets>
         <transnexus.com:Fraction critical="False">0</transnexus.com:Fraction>
          </transnexus.com:LossReceived>
        </transnexus.com:Statistics>
      </UsageIndication>
   </Message>
```

Step 4B. OSP UsageIndication from Destination

From destination gateway, 10.10.1.56, to OSP Nexus Server.

```
   <Message messageId="2597369534664" random="2857">
     <UsageIndication componentId="2597369532132">
       <Timestamp>2001-10-03T22:05:34Z</Timestamp>
       <Role>destination</Role>
       <TransactionId>4304187353840953091</TransactionId>
       <CallId encoding="base64">jPm35LeBEdWALdS27YGXCQ==</CallId>
       <SourceInfo type="e164">11111111111</SourceInfo>
    <SourceAlternate type="transport">[10.10.1.57]</SourceAlternate>
    <DestinationInfo type="e164">12222222222</DestinationInfo>
    <DestinationAlternate
type="transport">[10.10.1.56]</DestinationAlternate>
       <UsageDetail>
         <Service/>
         <Amount>21</Amount>
         <Increment>1</Increment>
         <Unit>s</Unit>
       </UsageDetail>
       <gric.com:TransactionStartTime critical="False">
              2001-10-03T22:05:12Z</gric.com:TransactionStartTime>
       <gric.com:TransactionStatus critical="False">
         <gric.com:TransactionCode critical="False">1016
                                              </gric.com:TransactionCode>
      <gric.com:Description critical="False">normal call clearing
                                              </gric.com:Description>
       </gric.com:TransactionStatus>
     </UsageIndication>
   </Message>
```

Step 5A. OSP Confirmation to Source

Acknowledgement message from OSP Nexus Server to source gateway 10.10.1.57.

```
   <Message messageId='2597352024114' random='18941'>
     <UsageConfirmation componentId='2597352025057'>
       <Timestamp>2001-10-03T22:05:34Z</Timestamp>
       <Status>
```

```
      <Description>SUCCESS</Description>
      <Code>200</Code>
    </Status>
  </UsageConfirmation>
</Message>
```

Step 5B. OSP Confirmation to Destination

Acknowledgement message from OSP Nexus Server to destination gateway, 10.10.1.56.

```
<Message messageId='2597369534664' random='18941'>
  <UsageConfirmation componentId='2597369532132'>
    <Timestamp>2001-10-03T22:05:34Z</Timestamp>
    <Status>
      <Description>SUCCESS</Description>
      <Code>200</Code>
    </Status>
  </UsageConfirmation>
</Message>
```

OSP Token Format

OSP interoperability between different VoIP devices is dependent on the formatting of the OSP token passed in the H.323 call set-up or SIP INVITE message from the source device to the destination device. The source device must format the OSP token so that the destination device can recognize and validate the OSP Token.

Annex D of ETSI TS 101 321 V2.1.1 defines OSP token object identifiers when the token is carried as part of a call signaling message of an ASN.1-based protocol. Cisco has implemented OSP tokens as clear tokens in an XML format. The osp-token-xml-format OID used by Cisco is: 040132112. A description of the Cisco OSP token is provided below.

Cisco OSP Token

1. Token should be present in the AuthorisationResponse message, one token for each Destination element.

2. Token uses XML format, with the following fields present:

   ```
   TokenInfoRandom:    provided by the settlement server
   SourceInfo:         same as in AuthorisationRequest
   DestinationInfo:    same as in AuthorisationResponse
   CallId:             same as in AuthorisationRequest
   TransactionId:      same as in AuthorisationResponse
   ValidUntil:         provided by the settlement server
   ValidAfter:         provided by the settlement server
   UsageDetail:        provided by the settlement server
   ```

3. Token is signed, but not encrypted, conforming to PKCS#7 standard for Signed Data.

   Section 7 in DTS03004 describes in details the necessary steps to create a signature.

```
    The signer certificate should be sent as part of
    the Signed Data content, in the "certificates"
    fields so that the router can verify the token
    without having to store the server's certificate.

The following is an example of the token in PKCS#7 signed data
format:

ContentInfo ::= SEQUENCE {
  contentType  { pkcs-7 signedData(2) }
  content      tokenSigned
}

tokenSigned SignedData ::= {
    version 1
    digestAlgorithms { iso(1) member-body(2) US(840) rsadsi(113549)
                     digestAlgorithm(2) 5}

    contentInfo {
        contentType { pkcs-7  1} -- data identifier
        content token            -- octet string representing the OSP
                                 -- token in XML format.  This token
                                 -- is created by the settlement server
    }

    certificates { -- settlement server certificate chain
        certificate {
            version 3
            serialNumber          -- the settlement server certificate
                                  -- serial number
            signature { pkcs-1 4 } -- md5WithRSAEncryption
            issuer                -- the certificate authority issuer name
            validity {
                notBefore         -- UTC time
                notAfter          -- UTC time
            }
            subject               -- the settlement server subject name
                                  -- as given in PKCS#10
            subjectPublicKeyInfo {
                algorithm { pkcs-1 1}
                subjectPublicKey  -- a BER encoding of the settlement server
                                     public key as given in PKCS#10
            extensions            -- the extentions as given in PKCS#10
            signatureAlgorithm { pkcs-1 4 }
            }
        }
        certificate               -- the certificate authority certificate
    }

    signerInfo  {  -- including the digest of the token as
                   -- the authenticated attributes
        version 1
        issuerAndSerialNumber {
            issuer                -- the certificate authority issuer name
            serialNumber          -- the CA's certificate serial number
        }
        digestAlgorithm { iso(1) member-body(2) US(840) rsadsi(113549)
                          digestAlgorithm(2) 5}

        authenticateAttributes {
            contentType { {pkcs-9 3} {pkcs-7 1}}
            messsageDigest { {pkcs-9 4}  -- an octet string }

        }

        digestEncryptionAlgorithm {pkcs-1 1}
        encryptedDigest "encrypted digest of the message using the
                         server private key"
```

```
    }
}
```

OSP token format for SIP

OSP, which is based on XML message transmitted via HTTP fits easily in the SIP architecture. OSP has been implemented have been implemented numerous SIP devices including the Vovida Policy Server, Commworks SIP proxy and Nuera's ORCA softswitch. As of this writing, the URL below references the IETF draft describing how an OSP token should be formatted in a SIP INVITE message:

http://www.ietf.org/internet-drafts/draft-johnston-sip-osp-token-02.txt

The title of the document is *OSP Authorization Token Header for SIP* written by A. Johnston, D. Rawlins, H. Sinnreich, and S. Thomas on October 23, 2001. This draft proposes a new SIP (Session Initiation Protocol) header OSP-Authorization-Token for carrying an OSP (Open Settlements Protocol) authorization token between domains.

A copy of this draft is included for your convenience in Appendix A.

Appendix A: *OSP Authorization Token Header for SIP*

Internet Engineering Task Force
Internet Draft
Document: draft-johnston-sip-osp-token-02.txt
October 2001
Expires: April 2002

A. Johnston
D. Rawlins
H. Sinnreich
WorldCom
Stephen Thomas
TransNexus

OSP Authorization Token Header for SIP

Status of this Memo

This document is an Internet-Draft and is in full conformance with all provisions of Section 10 of RFC2026[1].

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts. Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet- Drafts as reference material or to cite them other than as "work in progress."
The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt
The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

Abstract

This draft proposes a new SIP (Session Initiation Protocol) header OSP-Authorization-Token for carrying an OSP (Open Settlements Protocol) authorization token between domains.

1 Introduction

The problem of interdomain IP telephony calls with QoS is an important problem being addressed using AAA protocols. The new SIP [1] header proposed here is part of an approach to solving this problem, which is detailed in another draft [2].

2 Design Alternatives

The OSP Token is an opaque string to SIP which must be carried in the INVITE passed between domains. As such, the Token could be carried as a MIME attachment. However, there are three issues with this:

- Since the Token must be carried with the SDP, the INVITE would need to have a multipart MIME message body. If either User Agents do not support multipart MIME, the call will fail.

- The Token is used by both proxies and User Agents. As such, the proxy would have to decode the multipart MIME message body to extract the token. The general design of SIP is for message bodies to contain information of interest to end-points only, with information needed by proxies contained in headers.

- Multipart MIME encoding/decoding adds more delay to an already lengthy call setup procedure, as compared to header processing.

For these reasons, a new SIP header is proposed instead of a new MIME type for OSP authorization tokens.

Note that since OSP tokens are commonly constructed according to Cryptographic Message Syntax [3], their size may depend on the size of X.509 certificates embedded in the CMS format. In some cases the addition of a token may increase the size of a SIP INVITE datagram beyond the 576-byte or 1500-byte fragmentation limits. When such behavior is not desirable, it is recommended that systems use the abbreviated token format described in Annex D of [4].

3 Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [5] and indicate requirement levels for compliant SIP caller preferences implementations.

4 Header Field Definition

Table 1 specifies an extension of Table 5 in RFC 2543 [1] for the new header defined here.

	where	enc	e-e	ACK	BYE	CAN	INV	OPT	REG
OSP-Authorization-Token	R	n	h	-	-	-	o	-	-

Table 1: Summary of header field. The "where" column describes the request and response types with which the header filed can be used. The "enc" column describes whether this message header field MAY be encrypted end to end. "o": optional "-": not applicable, "R': request header, "r": response header, "g": general header, "*": needed if message body is not empty. A numeric value in the "type" column indicates the status code the header field is used with.

OSP-Authorization-Token = "OSP-Authorization-Token"":" Token
Token = quoted-string

6 Protocol Semantics

The OSP Token is always encoded per base64 and only allowed in INVITE requests and 200 OK responses to INVITEs.

6.1 User Agents

The UAC include the header an INVITE requesting QoS using AAA.

If it is absent in the INVITE, an AAA/QoS UAS determines the token and adds the header, otherwise it validates it.

6.2 Proxies

A proxy participating in the AAA exchange will examine and validate the token.

Otherwise, the header is ignored.

7 Example Message

```
INVITE sip:+1-972-555-5555@sip.domain2.com;user=phone SIP/2.0
Via: SIP/2.0/UDP sip.domain1.com:5060;branch=3a56d3.1
Via: SIP/2.0/UDP phone1.domain1.com:5060
From: Henry Sinnreich <sip:henry.sinnreich@phone1.domain1.com>
To: <sip:+1-972-555-5555@sip.domain2.com;user=phone>
Call-ID: 123456@domain1.com
CSeq: 1 INVITE
Contact: sip:henry.sinnreich@phone1.domain1.com
Record-Route: <sip:+1-972-555-5555@sip.domain2.com
 ;maddr=sip.domain1.com>
OSP-Authorization-Token:
_YT64VqpfyF467GhIGfHfYT6jH77n8HHGghyHhHUujhJh756t
HGTrfvbnjn8HHGTrfvhJhjH776tbB9HG4VQbnj7567GhIGfH
6ghyHhHUujpfyF47GhIGfHfYT64VQbnj_
Content-Type: application/sdp
Content-Length: 184

v=0
o=hsinnreich 9735285123 9721273312 IN IP4 122.32.11.6
s=Discussion of SIP QoS OSP for AAAArch
c=IN IP4 122.32.11.6
t=0 0
m=audio 9876 RTP/AVP 0
a=qos:mandatory recv confirm
```

8 Security Considerations

Since the AAA scheme [2] assumes authentication between client and server as well as IPSec AH between the server and the OSP server, the message and its contents (and therefore the token) can be trusted.

The token can also be encrypted.

No other security issues are introduced by this new header.

9 References

[1] M. Handley, H. Schulzrinne, E. Schooler, and J. Rosenberg, "SIP: session initiation protocol," Request for Comments (Proposed Standard) 2543, Internet Engineering Task Force, March 1999.

[2] H. Sinnreich, D. Rawlins, A. Johnston, S. Donovan, and S. Thomas, _AAA Usage for IP Telephony with QoS_ <draft-sinnreich-aaa-interdomain-sip-qos-osp-00.txt> Internet Draft, July 2000.

[3] R. Housley, "Cryptographic Message Syntax", RFC 2630, June 1999.

[4] European Telecommunications Standards Institute. "Telecommunications and Internet Protocol Harmonization Over Networks (TIPHON); Open Settlement Protocol (OSP) for Inter-domain pricing, authorization, and usage exchange". Technical Specification 101 321. Version 2.1.0.

[5] S. Bradner, "Key words for use in RFCs to indicate requirement levels," Request for Comments (Best Current Practice) 2119, Internet Engineering Task Force, March 1997.

Authors' Addresses

Alan Johnston
WorldCom
100 S. 4th Street
St. Louis, Missouri 63104
alan.johnston@wcom.com

Henry Sinnreich
WorldCom
400 International Parkway
Richardson, Texas 75081
USA
henry.sinnreich@wcom.com

Diana Rawlins
WorldCom
901 International Parkway
Richardson, Texas 75081

USA
diana.rawlins@wcom.com

Stephen Thomas
TransNexus
430 Tenth Street NW, Suite N204
Atlanta, GA 30318
USA
stephen.thomas@transnexus.com