# VoIP

## Voice over Internet Protocol

VoIP – v1.6 – 03/19/2008

(C) 2005,2012 Technologeeks.com - All Rights Reserved

# Table of Contents:



**The Technologeeks specialize in training and consulting services. This, and many other training materials, are created and constantly updated to reflect the ever changing environment of the IT industry.**

**To report errata, provide feedback, or for more details, feel free to email Info@Technologeeks.com**

**© Copyright 版權聲明**

**This material is protected under copyright laws. Unauthorized reproduction, alteration, use in part or in whole is prohibited, without express permission from the authors. You may distribute this freely, but not use it commercially.**

**We put a LOT of effort into our work (and hope it shows). Respect that.**

# Voice over Internet Protocol

Voice over the Internet Protocol (VoIP) heralds the next generation of Internet applications:

- Internet Telephony
- Video Conferencing (VVoip)
- Multimedia content on demand

We will discuss the commonly deployed VoIP technologies:

- H.323(v5): The older, stable protocol suite
- SIP: The new, emerging standard

Voice Over IP is the exciting prospect of replacing age old “PSTN” telephony with packet switched networks – first and foremost the internet. This is no longer limited only to voice – and now many multimedia protocols are used in the same way, including video.

We will focus on two contending standards:

**H.323** – The ITU.T standard, originally drafted to allow voice, and later expanded to general multimedia applications. This is a complex protocol suite, as we will demonstrate

**SIP** – The Session Initiation Protocol – a simpler, more generic approach, that is emerging as the chosen standard.

As well as mention Cisco’s custom implementation in older IP Phones, SCCP – The Selsius Call Control Protocol, affectionately called “Skinny”. This is a phased-out protocol, but still of some interest to us.

We Will NOT discuss other protocols, such as the H.248 (MegaCo), or The Multimedia Gateway Control Protocol (MGCP, specified in RFC2705). Nor will much detail be given to Skype – not to imply that it’s not common ; rather, that it is a closed protocol, of which little has been published.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

# H.323

(Probably The world's most convoluted protocol suite)

(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.323

# H.323

H.323 is an ITU-T Standard for voice, video and data over IP: "Packet-based multimedia communications systems".

Not a single protocol, but rather an "umbrella" specification:

H.323 brings together and defines the use of protocols for:

- **Signaling**: Call setup and maintenance
- **Media Transport**: Call switching over the IP framework
- **Media Encoding:** Codecs to encode/decode call data

The Internation Telecommunication Union Standardization Sector (ITU-T) are the standards body that introduce many communications standards. **http://www.itu.int/** - ITU-T's web site. ITU recommendations are classified by scopes, identified by letters of the alphabet. The scopes of interest to VoIP are:

**G** Transmission systems and media, digital systems and networks - Used in audio codecs. Specifically, we will see G.711, G.723 and G.729, among others.

**H** Audiovisual and multimedia systems – The H.323 family, including H.225 (RAS and CS), as well as video codecs (H.261-264),

**Q** Switching and signalling – Q.931 and Q.932

**T** Terminals for telematic services – T.120

(you might also recognize the letters **X** (X.25, X.500, X.509) and **V** (V.90, for past modems).

H.323 was first approved in February 1996. Since then it has undergone many revisions, and the current standard is H.323v5. It was the the first standards-based "Voice over IP, and enjoyed widespread use – although lately it has been losing grounds to SIP.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.323

# H.323 Devices

H.323 Devices can be categorized as follows:

- Terminals (Endpoints): providing UI, and duplex sessions in real time.
- Gateways: bridge between H.323 and non-H.323 (e.g PSTN) devices
- Gatekeepers: provide call control features, e.g.
  - Bandwidth Management
  - Address Lookup
- Multipoint Control Units (MCUs): enable conferencing

**<u>Terminals</u>**: are the endpoints of H.323 based communication, and the terms are often used interchangeably. These can be IP-Phones (with or without video capabilites), voice or video conferencing software (e.g. Netmeeting/Skype-types), Video recording devices, voice recording (e.g. voicemail) systems, and more.

**<u>Gateways</u>** bridge between H.323 and other systems. These may be:

- <u>PSTN:</u> Plain switched Telephone Networks – still common, but destined to become legacy
- <u>H.320 systems:</u> a precursor to H.323, still supported in some scenarios
- <u>Other H.323 systems:</u> in which case the gateway serves as an H.323 Proxy

Gateways consist of a "Media Gateway" (MG – to handle media translation issues, mostly codec issues), and "Media Gateway Controllers" (MGC - to handle signaling).

**<u>Gatekeepers</u>** provide many important logistic features in the H.323 network. These include:

- <u>Bandwidth Management</u>: ensuring QoS for voice and video sessions
- <u>Address Lookup</u>: Translating named addresses (such as phone numbers) into IP addresses
- <u>Admission Control</u>: Registration, Admission and Status (**RAS**) messages

Gatekeepers are actually optional components, and H.323 networks can operate fine without them. If they exist, however, the above functions are mandatory. Additionally, they may provide optional functions, such as:

- <u>Call Authorization</u>: Allowing/Rejecting calls for any reason
- <u>Call Management</u>: Rerouting calls to the next available terminal, voice mails, etc.
- <u>Call Signaling</u>: Acting as a proxy for two endpoints.

**<u>Multipoint Control Units</u>** (also referred to as **<u>MCU</u>**s) enable multipoint conferencing. These units contain Multipoint Controllers (MCs) that handle the conference responsibilities – mixing media from multiple sources, switching, etc.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.323

## H.323 Vendors

H.323 Is commonly employed in the following products

- Microsoft's NetMeeting
- Avaya's Multivantage
- Nortel's Meridian IP-PBX systems
- Cisco's Call Manager
- RadVision (H.323 stacks, GK, GWs)
- Asterisk (open source PBX)

H.323 is rapidly losing grounds to SIP, but still common

H.323 was the first VoIP protocol suite, and is implemented in several products, most notably Microsoft's "NetMeeting" software. As we will see, H.323 is losing grounds to SIP. SIP, unlike H.323 is an open protocol, and textual.

**http://www.h323forum.org/products/** has a better list of exactly who/what/why supports H.323.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.323

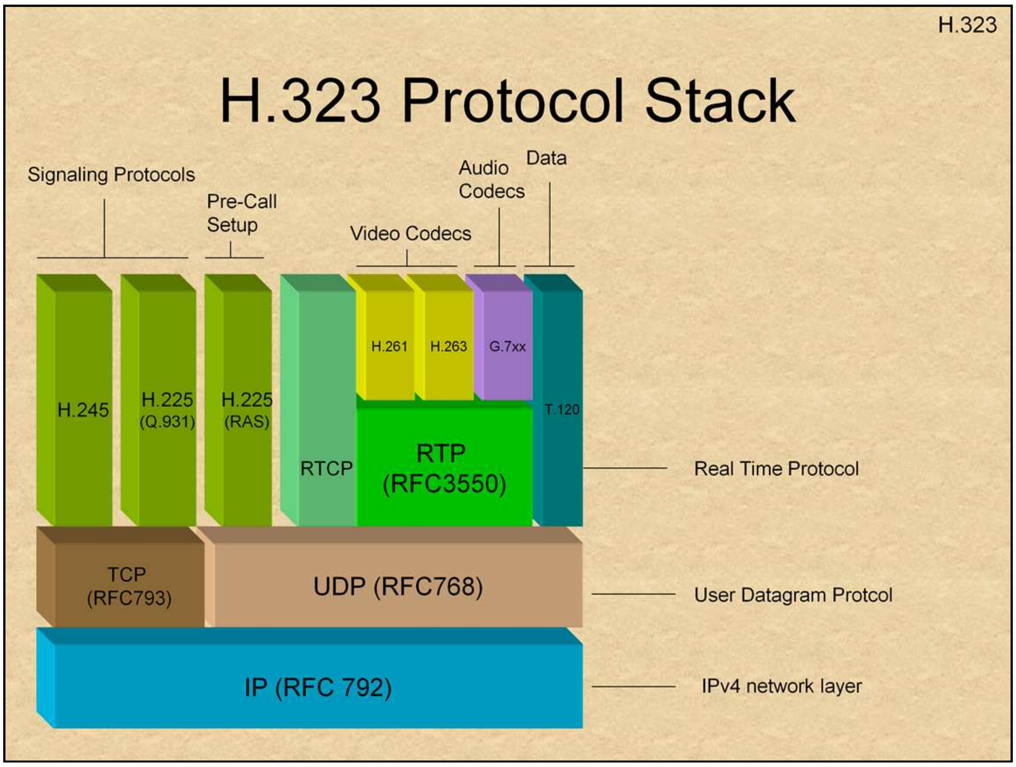
# H.323 Versions

H.323 has evolved considerably over the years:

| Version | Year | Major Changes/Addenda |
|---|---|---|
| H.323v2 | 1998 | - Fast Connect<br>- H.245 Tunneling<br>- Unique Call Identifiers (GUIDs)<br>- H.235 Security Additions |
| H.323v3 | 1999 | -Reusing signaling connections<br>- H.341 – SNMP MIB<br>- H.282 – Remote Device Control |
| H.323v4 | 2000 | -H.248 introduced<br>- H.245 Parallel to Fast Connect<br>- H.450.x – Call completion/Offer/Intrusion<br>- RFC 2833 (DTMF) |
| H.323v5 | 2003 | H.460.x – Generic Extensibility Framework |
| H.323v6 | 6/2006 | H.235.x – Security revamped<br>H.460.10-21 – including NAT traversal |

**http://www.packetizer.com/** has an excellent reference on the differences between H.323 protocol versions. The exact URL is **http://www.packetizer.com/voip/h323/whatsnew_v?.html** – replacing '?' with [2-6]. The slide above lists the major changes in each protocol version, rather comprehensive, but not complete.

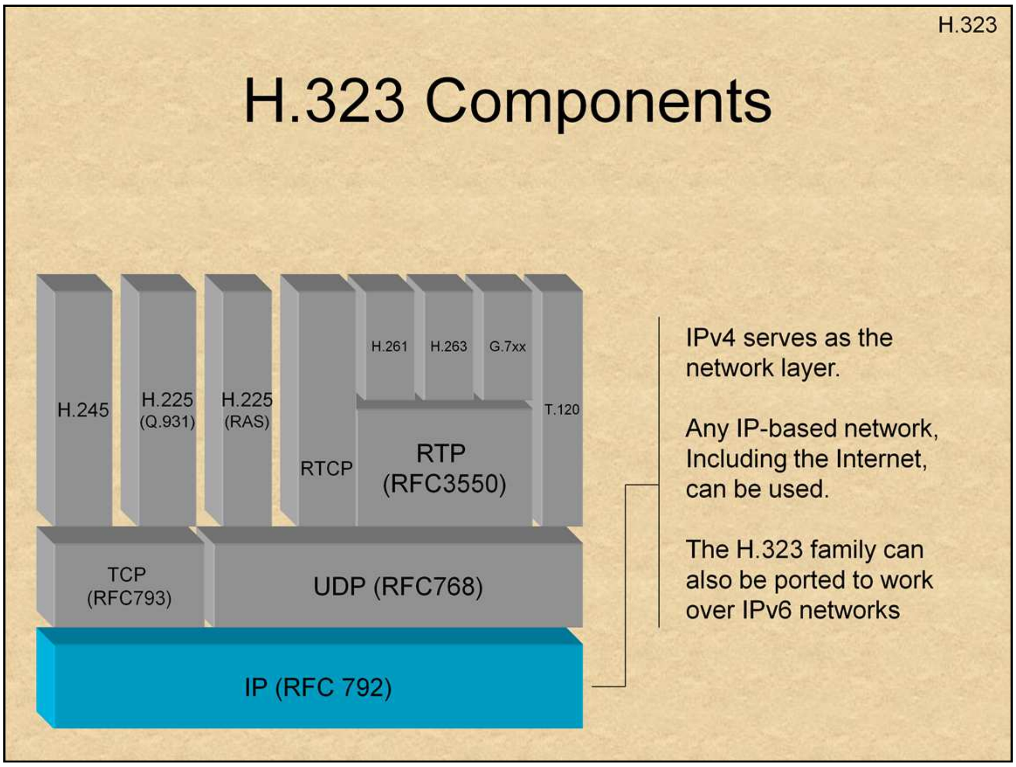
(C) 2005,2012 Technologeeks.com - All Rights Reserved




The above slide shows the H.323 “Protocol Stack” – and is an important illustration as to how all the various building blocks of H.323 “fit together”.

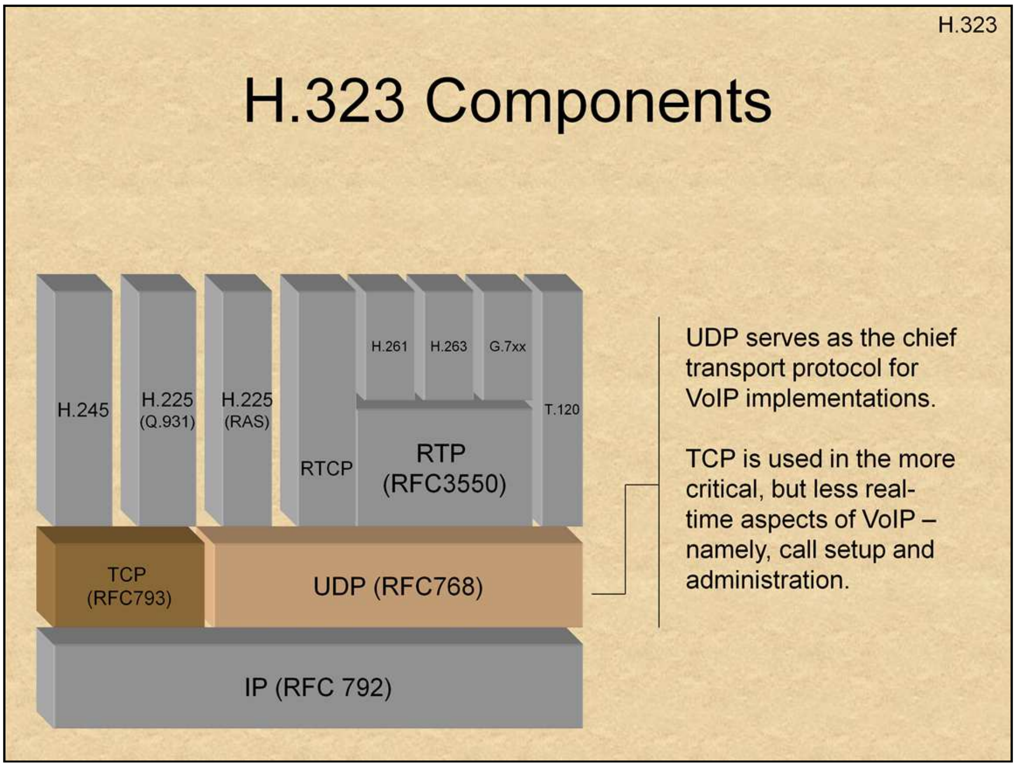
We next consider the H.323 Components, one by one.

(C) 2005,2012 Technologeeks.com - All Rights Reserved




IPv4 serves as the basic transport for Voice over IP – no surprise there. Recent implementations of H.323 also support IPv6, as well.

(C) 2005,2012 Technologeeks.com - All Rights Reserved



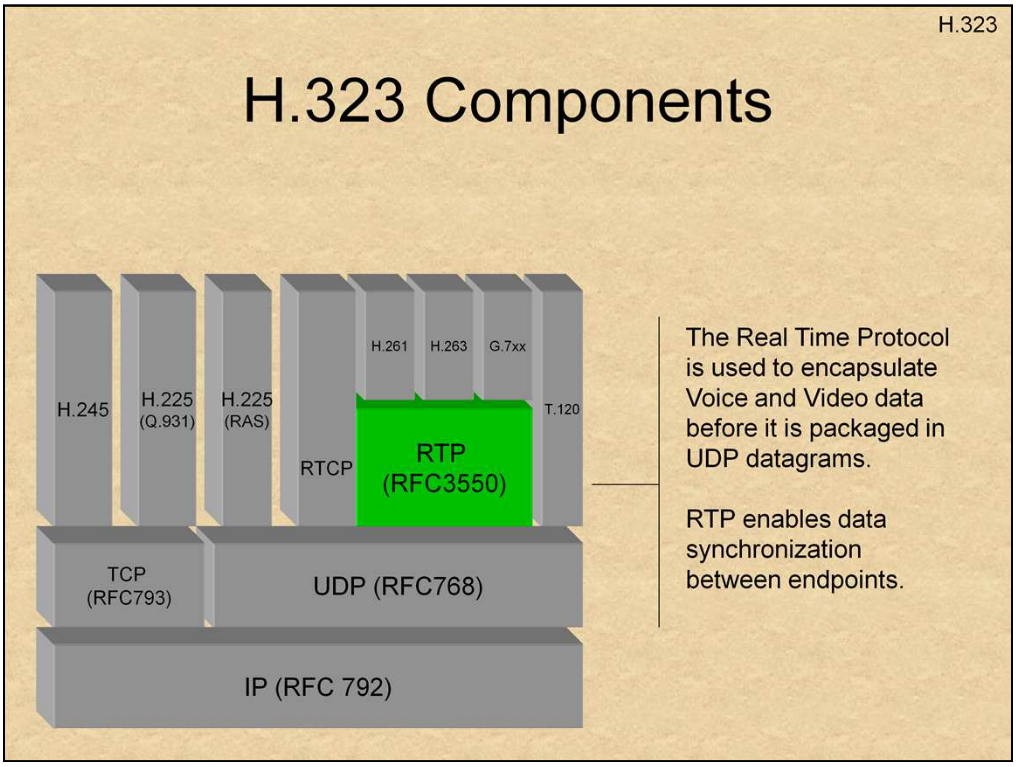

Contrary to many other protocols, VoIP primarily used UDP, and not TCP. While TCP is, by far, the more reliable of the two, it is incompatible with the Real-Time requirements of VoIP. This is due to TCP's slow nature, resulting from its acknowledgements:

- Real-Time performance does not allow for acknowledgements. In case packets are dropped, it's better to suffer a glitch in the voice or video stream, rather than freeze the connection until the packets are retransmitted.

- Multi-point conferences also cannot work with acknowledgements – no sense holding everyone back because one specific recipient has not obtained packets within a reasonable time.

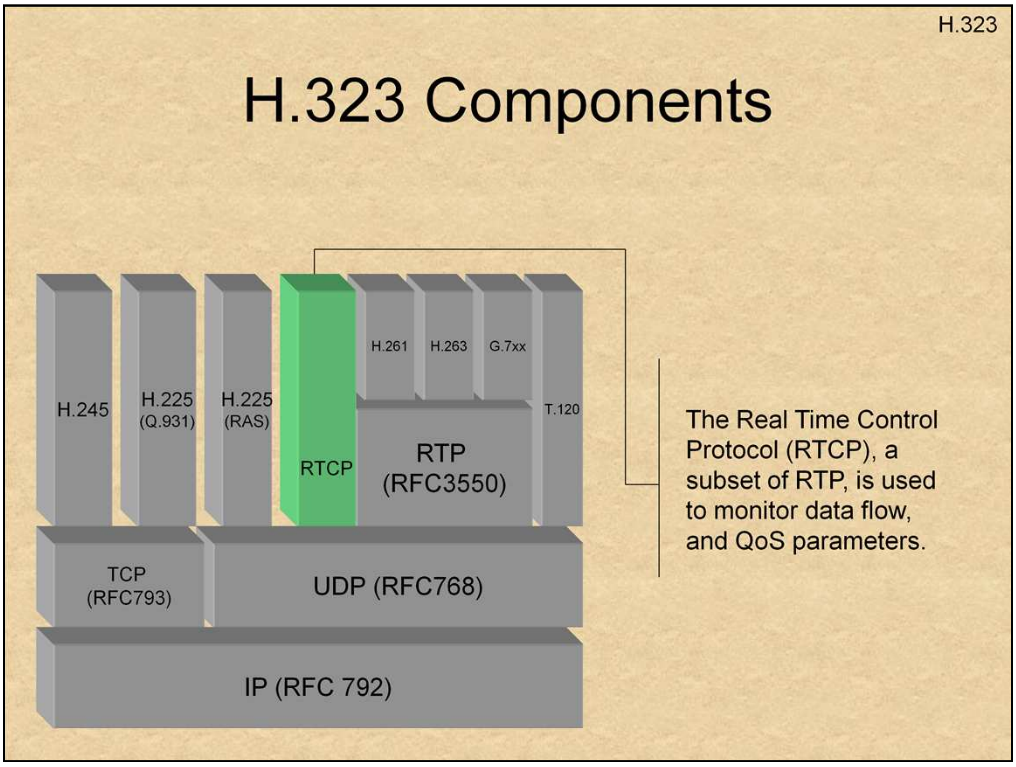
TCP is still used where reliability is valued over performance. This is in most operations where real time performance is not required – such as call setup, control, and signaling.

(C) 2005,2012 Technologeeks.com - All Rights Reserved



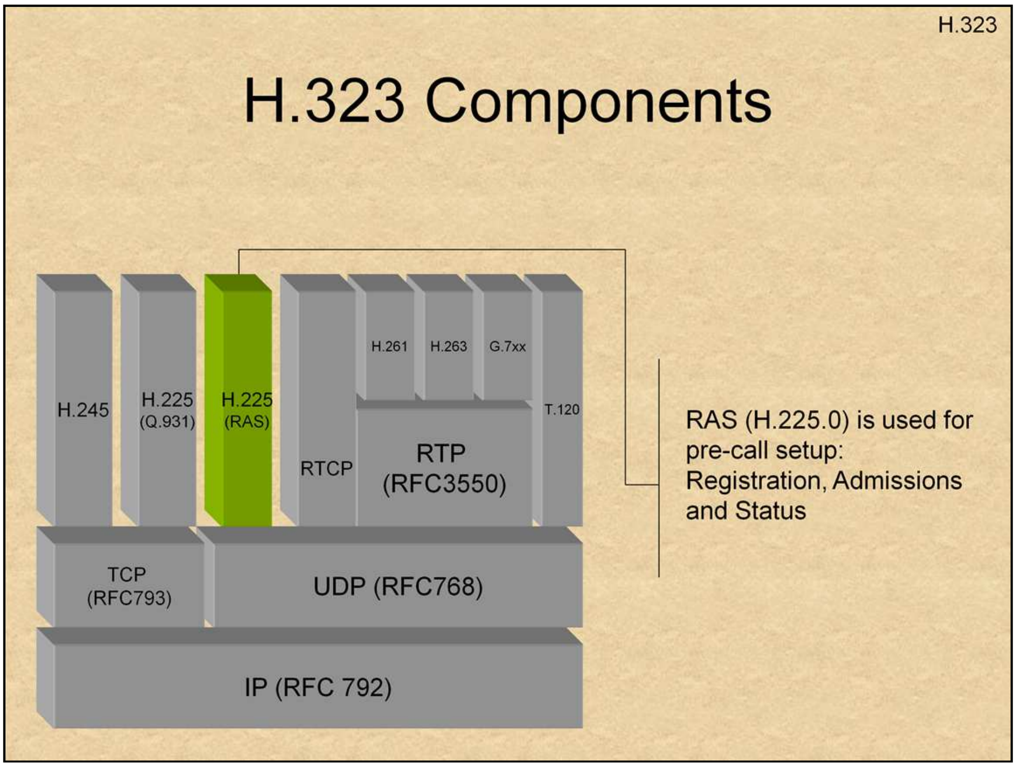

A key transport protocol used in H.323 is RTP – The Real Time Protocol (RFC3550). This protocol will be elaborated in greater detail in a few pages.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

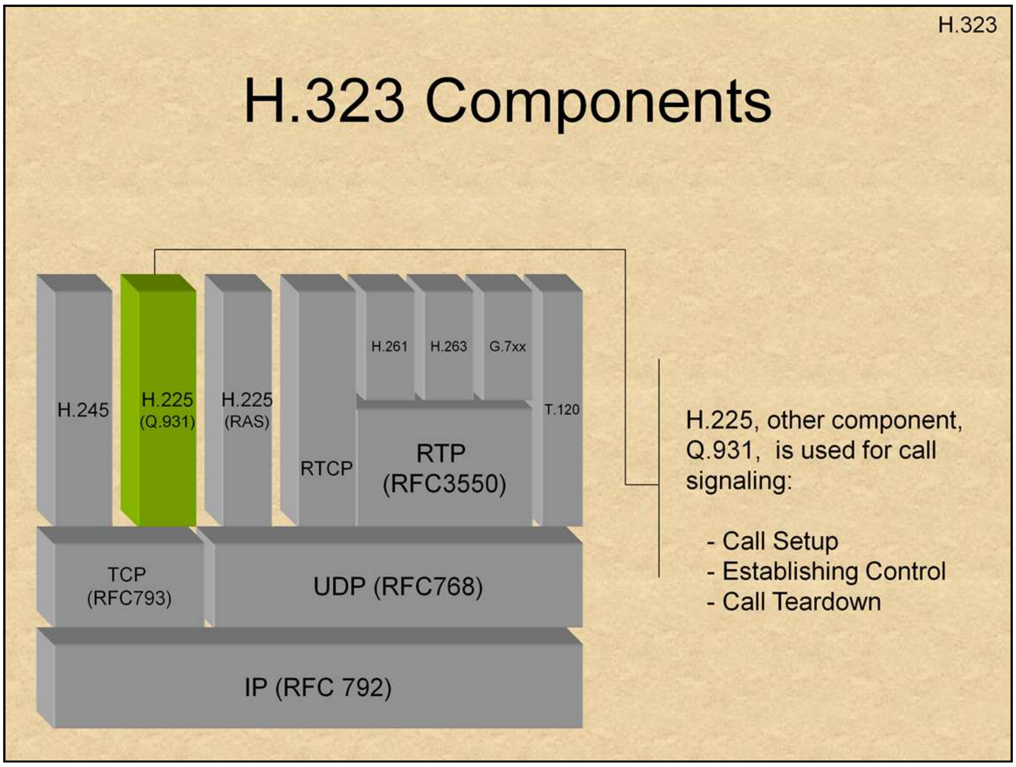



RFC3550, the RTP specification, also introduces the Real Time Control Protocol – RTCP. This is essentially a subset of RTP, used to provide real-time statistics on data flow, so Quality of Service can be measured in real time. RTCP will also be discussed in more detail, shortly.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

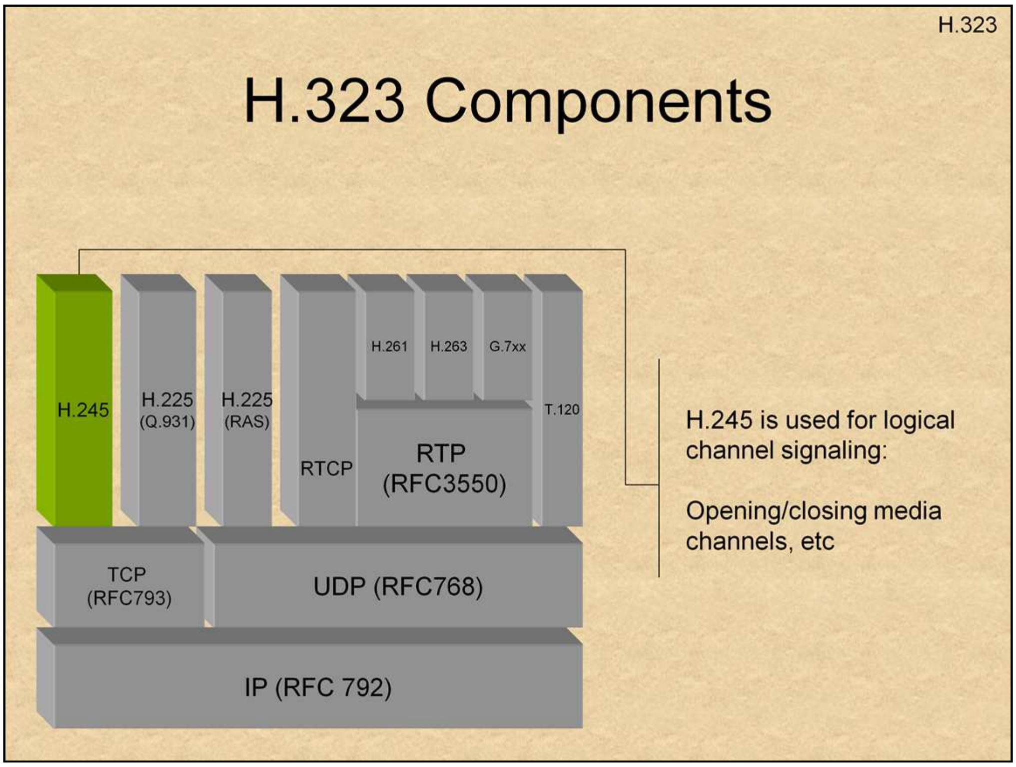



RAS is a sub-component of H.225 used for the pre-call setup stages, as well as modifying call parameters.



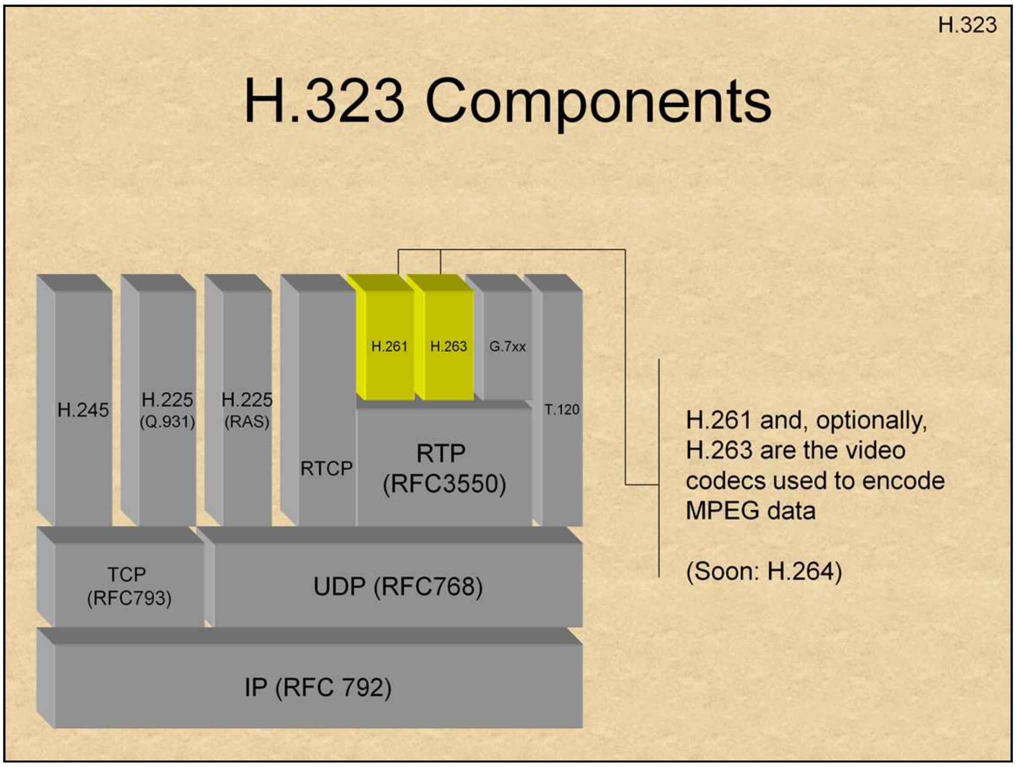

H.225.0 – “Call signalling protocols and media stream packetization for packet-based multimedia communication systems”

(C) 2005,2012 Technologeeks.com - All Rights Reserved




H.245 – "Control protocol for multimedia communication" defines the protocol that is used for logical channel control, or signaling.  This involves setting up the transport channels for the various media streams.

(C) 2005,2012 Technologeeks.com - All Rights Reserved



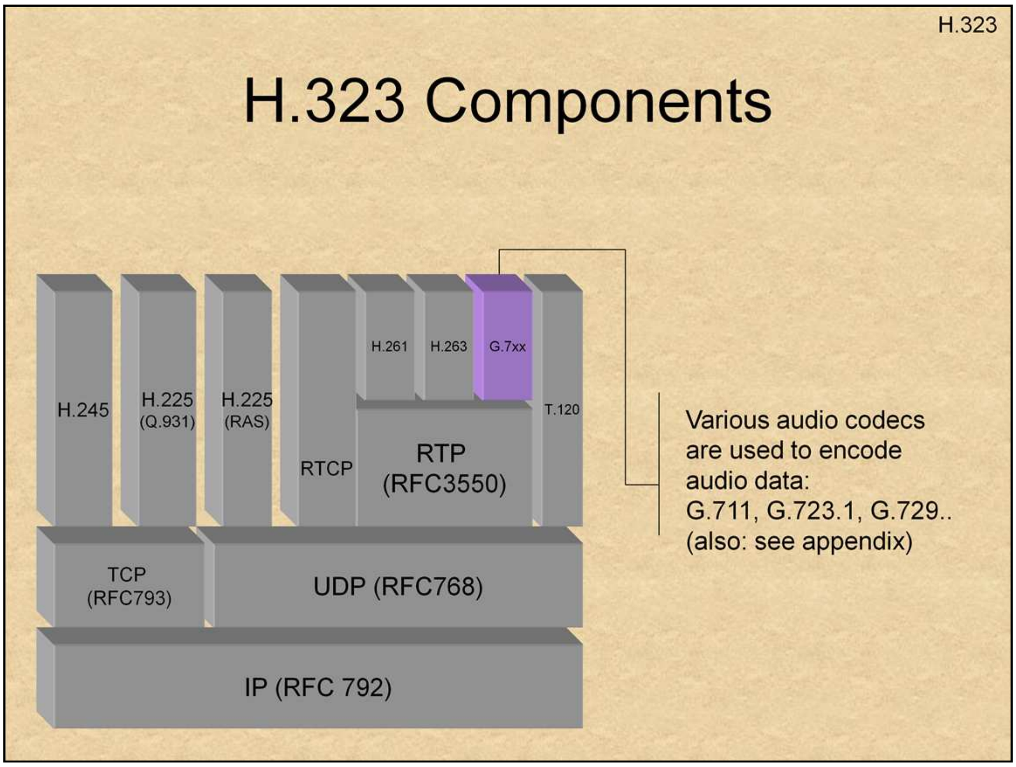

The H.2xx standards are codecs for transmitting video data:

**H.261** - Video codec for audiovisual services at p x 64 kbit/s (high bit-rate, VHS Quality)

**H.263** - Video coding for low bit rate communication - Supports common interchange format (CIF), quarter common interchange format (QCIF), and sub-quarter common interchange format (SQCIF) picture formats and is superior for Internet transmission over low-bit-rate connections..

Fairly recently, a new video codec, H.264, has made an appearance. This is a very high quality codec used in high-end video conferencing, comparable to DiVX/XViD.

(C) 2005,2012 Technologeeks.com - All Rights Reserved




Similarly, G.7xx standards are used for audio data: ***(for more on this, see appendix).*** Codecs are rated by a Mean Opinion Score – MOS, ranging from 1 (Bad) to 5 (Excellent) – according to the ITU standard P.800.

**G.711** - Pulse code modulation (PCM) of voice frequencies - 64Kbps – default for PSTN PBXes. Its MOS score is the highest, 4.1

**G.723** - Dual rate speech coder for multimedia communications transmitting at 5.3 (ACELP – MOS 3.65) or 6.3 (MP-MLQ - MOS 3.9) kbit/s.

**G.726** - 40, 32, 24, 16 kbit/s adaptive differential pulse code modulation (ADPCM)

**G.728** - 16 kbit/s Low Delay Code Excited Linear Prediction (LD-CELP).

**G.729** - Coding of speech at 8 kbit/s using conjugate-structure Algebraic-Code-Excited Linear-Prediction (CS-ACELP). Skype has been known to use this. Quality is as good as 32Kbps adaptive PCM. A variant (G.729a) is slightly less performant. Its MOS score is 3.92

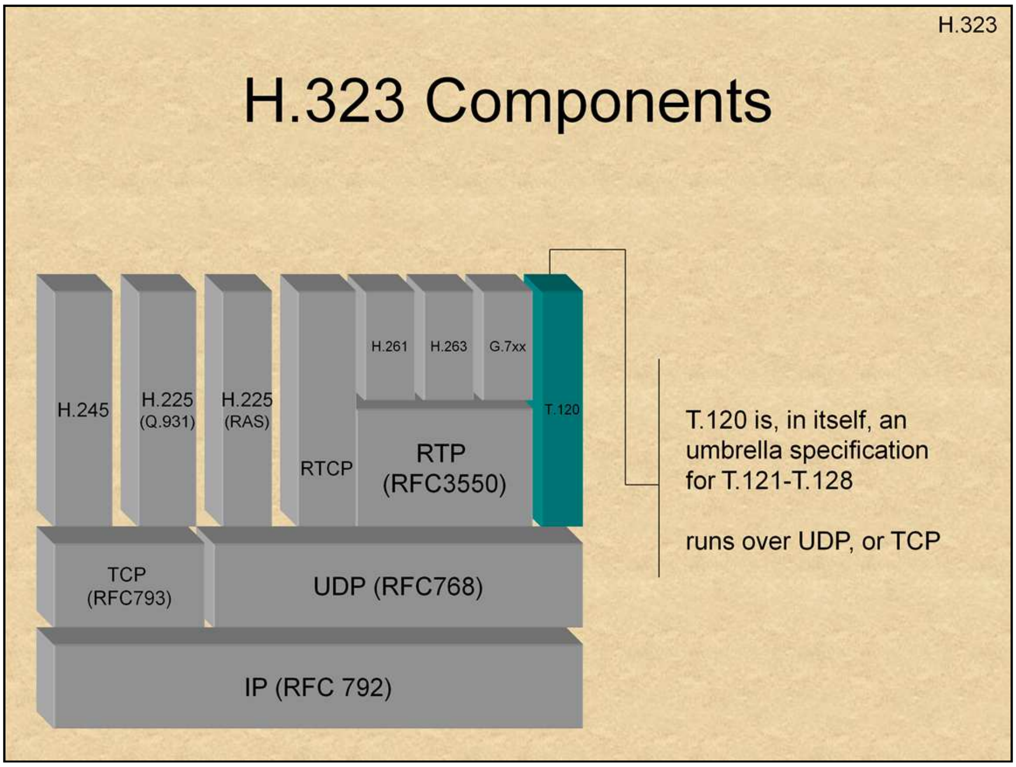
**iLBC** - A freeware codec optimized for Internet telephony (Internet Low Bitrate Codec) – Encoding frame length can be 20ms (resulting 15.20Kbps) or 30ms (resulting in 13.33). Used extensively in many applications and softphones/messengers, as well as, allegedly, PacketCable.

GlobalIPSound (http://www.globalipsound.com/solutions/solutions_Codecs.php) licenses many other codecs, especially those used by Skype.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

| name of encoding | sample/frame | bits/sample | sampling rate | ms/frame | default ms/packet |
|---|---|---|---|---|---|
| G711 | sample | 8 | 8,000 | | 20 |
| G722 | sample | 8 | 16,000 | | 20 |
| G723.1 | frame(1) | N/A | 8,000 | 30 | 30 |
| G726-40 | sample | 5 | 8,000 | | 20 |
| G726-32 | sample(2-20) | 4 | 8,000 | | 20 |
| G726-24 | sample | 3 | 8,000 | | 20 |
| G726-16 | sample | 2 | 8,000 | | 20 |
| G728 | frame(4-64) | N/A | 8,000 | 2.5 | 20 |
| G729 | frame(2-64) | N/A | 8,000 | 10 | 20 |
| G729D | frame | N/A | 8,000 | 10 | 20 |
| G729E | frame | N/A | 8,000 | 10 | 20 |
| GSM | frame | N/A | 8,000 | 20 | 20 |
| GSM-EFR | frame | N/A | 8,000 | 20 | 20 |
| G.711a/u | sample | 8 | var. | | 20 |
| ISAC | frame | 10-32Kbps | 16,000 | | 30-60 (Used by Skype) |
| iLBC | frame | | | 20 or 30 | 15.2kbps/13.3Kbps |

(C) 2005,2012 Technologeeks.com - All Rights Reserved




**T.120** is used for application data transfer over the H.323 framework. It is an umbrella specification for the following sub-protocols:

**T.121** The Generic Application Template (GAT), serves as:

- A framework for the other application protocls
- Registration/deregistration
- Capability determination and negotiation

**T.122** – Multipoint services

**T.123** – Sequencing, error-correcting and transporting data. Annex B specifies secure conferencing.

**T.124** – Generic Conference Protocol

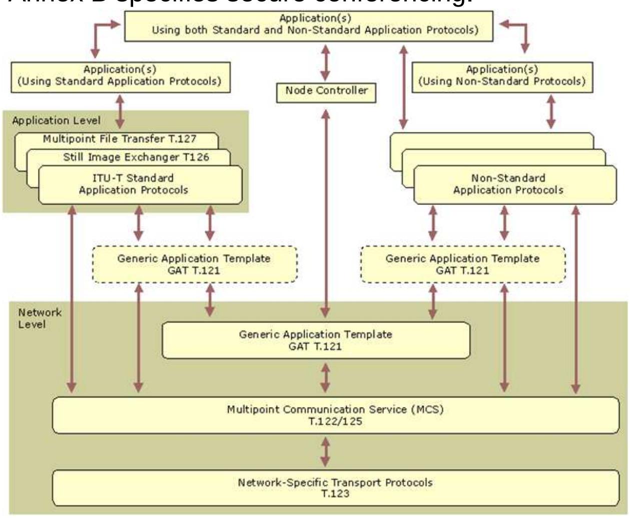
**T.125** – Data Channels in conference

**T.126** – “Whiteboard” application sharing

**T.127** – File Transfers

**T.128** – A Microsoft extension used in NetMeeting’s application sharing and collaboration

This diagram, taken from Microsoft’s NetMeeting Resource Kit*, shows the complexity of T.120.




* - http://www.microsoft.com/windows/NetMeeting/Corp/reskit/chapter10/default.asp

(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.323

# H.323 Call Flow

**Precursor step:**
- Registration via RAS of endpoints

**Call Setup:**
- "DNS"-Like Lookup of Address
- H.225: RAS – Request for authorization?
- H.225: Emulate phone "Ringing" and "Pick-Up"

**Call Flow:**
- H.245 Logical Channel Setup, Capability negotiation
- RTP Data flow in opened logical channels ensues

H.323 defines a lengthy process to setup a multimedia call. The process consists of the following stages as shown above. We will now focus on each of these stages.

H.225

# H.225.0

- Handles Call Setup and Signaling
  - Call Connection
  - Call Maintenance
  - Call Disconnection
- Composed of two components:
  - RAS – Registration, Admission, Status (UDP)
  - Q.931/Q.932 – call setup/control (TCP)

TCP port 1720 is generally the reserved port for H.225.0 messages. The protocol is flexible enough, however, to move to ephemeral ports during a call.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

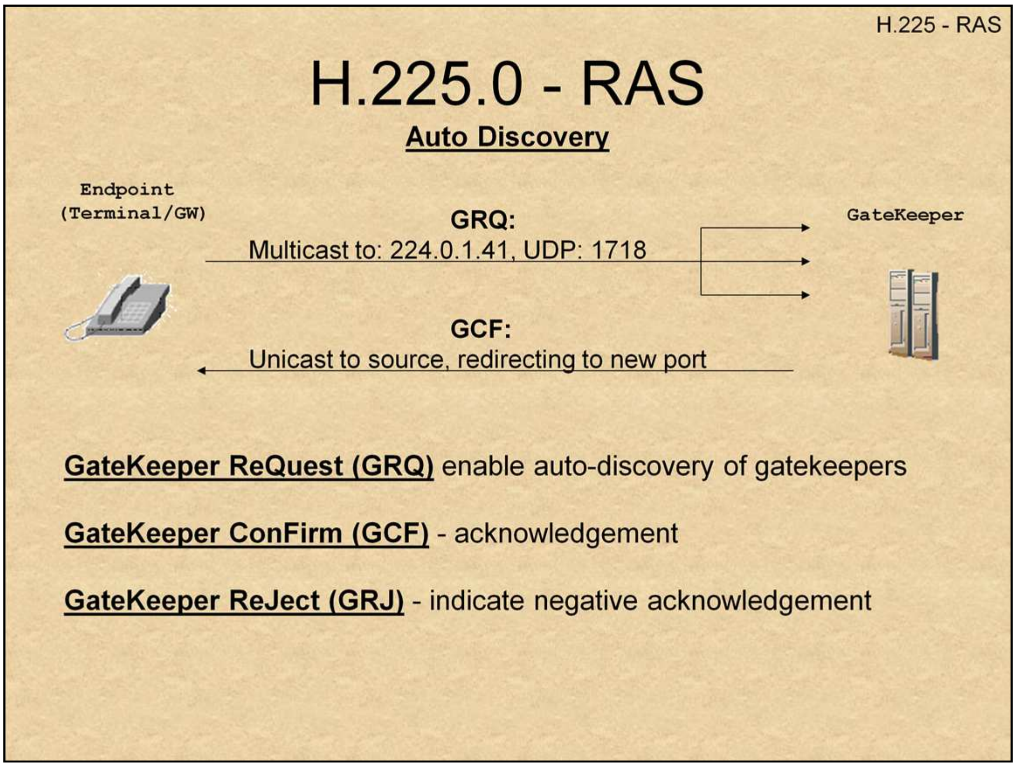
H.225 - RAS

# H.225.0 - RAS

- H.225 subprotocol for Registration, Admissions, Status
- Gatekeeper discovery:
  - UDP dest 1718, Multicast dest 224.0.1.41
- Registration/Unregistration messages, etc:
  - UDP dest 1719, Unicast

Gatekeeper Discovery is a strictly optional feature of H.323. Systems with no multicast support, such as Microsoft NetMeeting, would require a hard coded GateKeeper value, as is shown in this dialog box, from Microsoft NetMeeting's "Advanced Calling Options".


(C) 2005,2012 Technologeeks.com - All Rights Reserved




The ASN.1 syntax of GRQ is shown below:

```
GatekeeperRequest ::= SEQUENCE
{
  requestSeqNum RequestSeqNum,
  protocolIdentifier ProtocolIdentifier,
  ...,
  rasAddress TransportAddress,
  endpointType EndpointType,
  gatekeeperID GatekeeperIdentifier OPTIONAL,
  endpointAlias SEQUENCE OF AliasAddress OPTIONAL,
  ...,
  tokens SEQUENCE OF ClearToken OPTIONAL,
  cryptoTokens SEQUENCE OF CryptoH323Token OPTIONAL,
  authenticationCapability
  SEQUENCE OF AuthenticationMechanism
  OPTIONAL,
  ...
}

GatekeeperConfirm ::= SEQUENCE
{
  requestSeqNum RequestSeqNum,
  gatekeeperID GatekeeperIdentifier OPTIONAL,
  rasAddress TransportAddress,
  ...,
  alternateGatekeeper SEQUENCE OF AlternateGK OPTIONAl,
  authenticationMode AuthenticationMechanism OPTIONAL,
  tokens SEQUENCE OF ClearToken OPTIONAL,
  cryptoTokens SEQUENCE OF CryptoH323Token OPTIONAL,
}
```

(Full ASN.1 specs may be obtained from http://www.packetizer.com/voip/h323/h2250v1.asn)

(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.225 - RAS

# H.225.0 - RAS

## Registration

Endpoint (Terminal/GW) GateKeeper

**RRQ:**
Unicast to RAS, to RAS defined port

**RCF:**
Unicast to source, acknowledging registration

**Registration ReQuest (RRQ)** messages are used to add the device to the Gatekeeper zone. Reply may be RCF (Confirmation) or RRJ (Reject).

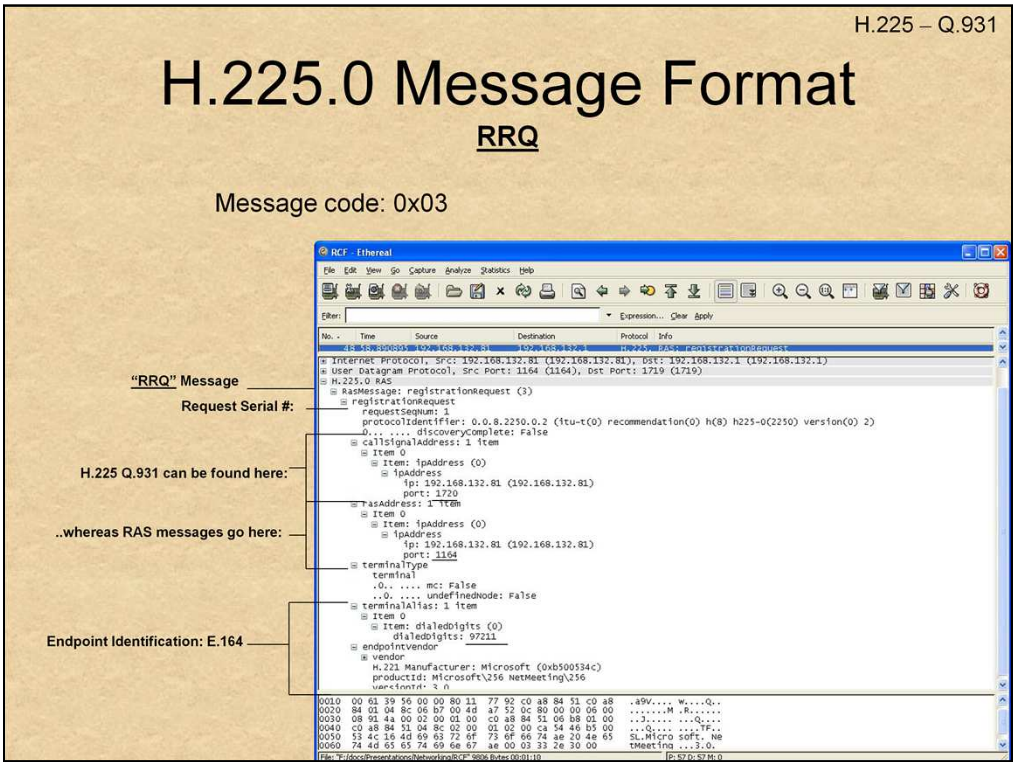
Similarly, **Unregistration ReQuests** are used for leaving zone (**URQ/UCF/URJ**)

To register, a device supplies one of two:

- **H.323 ID:** An email address "account"
- **E.164 Address:** A.K.A a phone number.

A RCF confirms the device is registered, and may be found by entities looking it up (see next page).


(C) 2005,2012 Technologeeks.com - All Rights Reserved


H.225 – Q.931
H.225.0 Message Format
RRQ
Message code: 0x03
"RRQ" Message
Request Serial #:
H.225 Q.931 can be found here:
..whereas RAS messages go here:
Endpoint Identification: E.164
RCF - Ethereal
File Edit View Go Capture Analyze Statistics Help
Filter: Expression... Clear Apply
No. Time Source Destination Protocol Info
Internet Protocol, Src: 192.168.132.81 (192.168.132.81), Dst: 192.168.132.1 (192.168.132.1)
User Datagram Protocol, Src Port: 1164 (1164), Dst Port: 1719 (1719)
H.225.0 RAS
RasMessage: registrationRequest (3)
registrationRequest
requestSeqNum: 1
protocolIdentifier: 0.0.8.2250.0.2 (itu-t(0) recommendation(0) h(8) h225-0(2250) version(0) 2)
0... .... discoveryComplete: False
callSignalAddress: 1 item
Item 0
Item: ipAddress (0)
ipAddress
ip: 192.168.132.81 (192.168.132.81)
port: 1720
rasAddress: 1 item
Item 0
Item: ipAddress (0)
ipAddress
ip: 192.168.132.81 (192.168.132.81)
port: 1164
terminalType
terminal
.0.. .... mc: False
..0. .... undefinedNode: False
terminalAlias: 1 item
Item 0
Item: dialedDigits (0)
dialedDigits: 97211
endpointVendor
vendor
H.221 Manufacturer: Microsoft (0xb500534c)
productId: Microsoft\256 NetMeeting\256
0010 00 61 39 56 00 00 80 11 77 92 c0 a8 84 51 c0 a8 .a9V.... w....Q..
0020 84 01 04 8c 06 b7 00 4d a7 52 0c 80 00 00 06 00 .......M .R......
0030 08 91 4a 00 02 00 01 00 c0 a8 84 51 06 b8 01 00 ..J..... ...Q....
0040 c0 a8 84 51 04 8c 02 00 01 02 00 ca 54 46 b5 00 ...Q.... ....TF..
0050 53 4c 16 4d 69 63 72 6f 73 6f 66 74 ae 20 4e 65 SL.Micro soft. Ne
0060 74 4d 65 65 74 69 6e 67 ae 00 03 33 2e 30 00 tMeeting ...3.0.
File: "F:/docs/Presentations/Networking/RCF" 9806 Bytes 00:01:10
P: 57 D: 57 M: 0

H.225 - RAS

# H.225.0 - RAS

**Endpoint Lookup**

Endpoints are located by the following messages:

**Locate Request (LRQ)** – Request a lookup of one or more E.164 addresses (i.e. phone #) or email addresses (H.323 IDs).

**Locate Confirm (LCF)** – Contains address of endpoint, or gatekeeper router.

**Locate Reject (LRJ)** – Node not registered, or incapable of handling sessions.

Location Requests are the H.323 equivalent of DNS lookups. The Lookup can be performed by one of two identifiers, as was shown on the last page – either the E.164 ID (Phone Number), or the H.323 ID (an E-Mail address).

The Gatekeeper therefore doubles as a name resolver. This is a most common configuration, as in most cases the endpoints do not have any name resolution capability

(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.225 - RAS

# H.225.0 - RAS

## Admissions

Calls are set up by the following messages:

**Admission Request (ARQ)** – Request for call setup, and bandwidth reservation

**Admission Confirm (10 - ACF)** – Bandwidth allocation, and address of remote gateway

**Admission Reject (11 - ARJ)** – Rejects call setup.

Likewise, DisengageRequests (DRQ, 16 - DCF, DRJ) hang up calls.

The Admission messages are used to query the gatekeeper, if present, to allow the call to proceed. This can involve validating the call vs. the local administrative policy, reserve proper bandwidth, etc. Similar to the Locate requests, they may be confirmed or rejected.

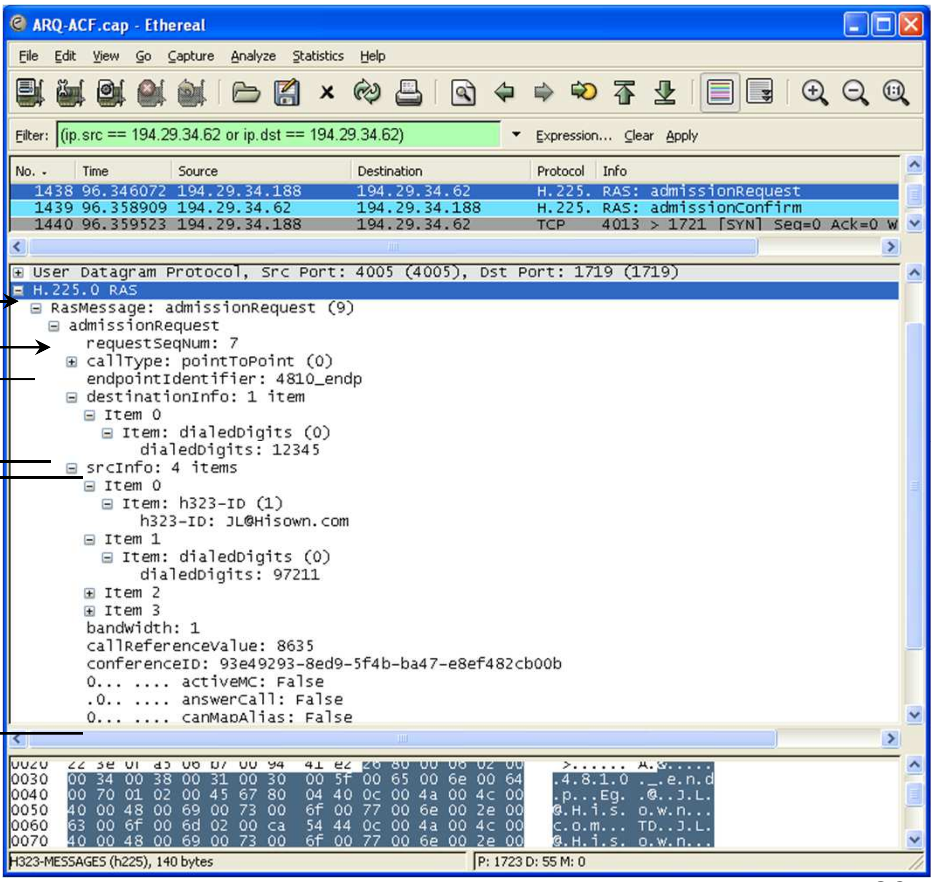
ARQs save the endpoint the need for LRQs, since they contain the address of the call recipient.

Message Type 9 - ARQ

Request for Peer-To-Peer call

CallER identifier. Note this gets the GateKeeper to perform the LCF, implicitly, thereby saving a separate req.

CallER identifiers, as well as globally Unique Conference and call IDs for this Call (call ID not shown)



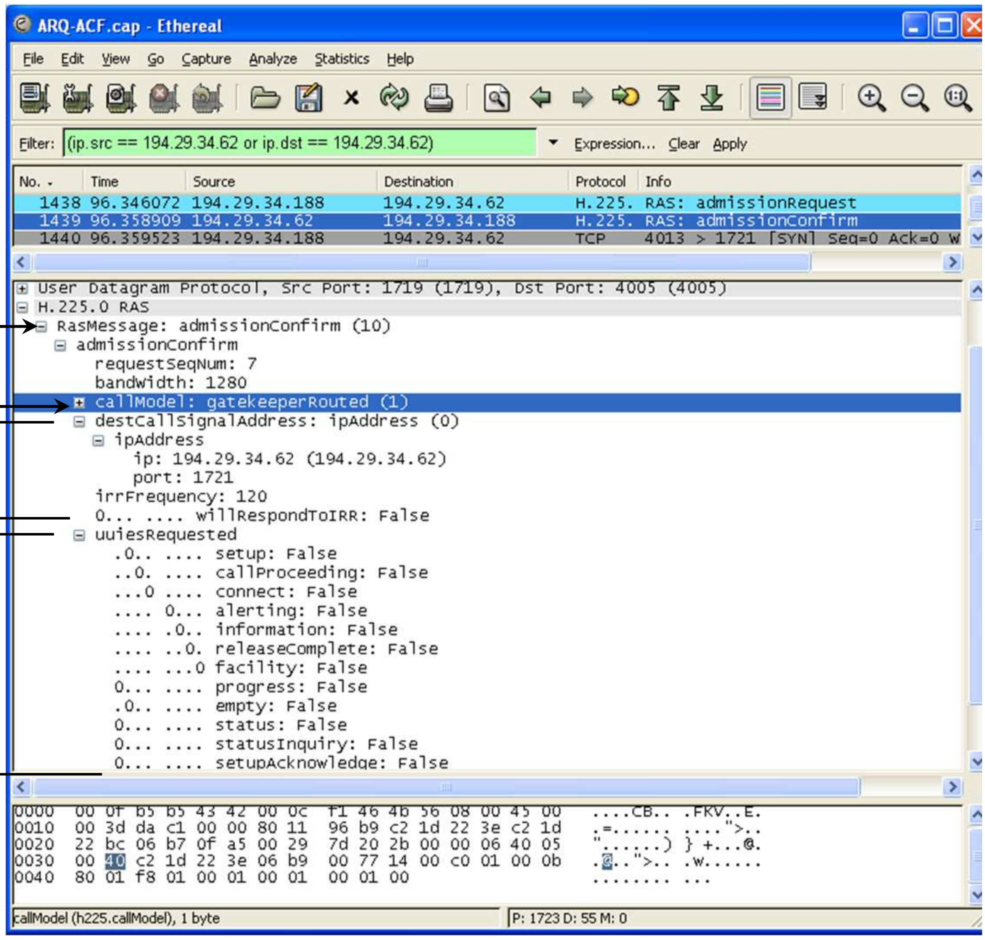
The ACF is shown here. Note that even though we requested a point-to-point call (callModel 0) the GateKeeper will only allow a GateKeeper Routed call (callModel 1). This means that the destCallSignalAddress provided is that of the GateKeeper, NOT the endpoint. (In this particular case, TCP port 1721 of the GateKeeper).

Message Type 10 - ACF

GateKeeper will route the call

Redirecting call to gatekeeper's TCP port 1721. Note that next packet is indeed a TCP SYN to 1721

Information elements requested






(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.225 - RAS

# H.225.0 - RAS

## Status

Status notification messages:

**Information ReQuest (21 - IRQ)** – Gatekeeper to endpoint, requesting status and "pinging" status

**Information Request Response (IRR)** – Sent from endpoint to gatekeeper, either as reply, or periodically.

**InfoRequestAck (IACK) or InfoRequestNack (INAK)** – confirm IRRs

**Status Enquiry (SEQ)** – Sent on call-channel, to verify calls are still active

Status notification Information Requests (IRQ) are sent in the opposite direction – from GateKeeper to endpoint. These requests solicit the response from the Endpoint. The endpoint then responds with an IRR. The GateKeeper may then acknowledge (IACK) or negative-acknowledge (INAK) the IRR.

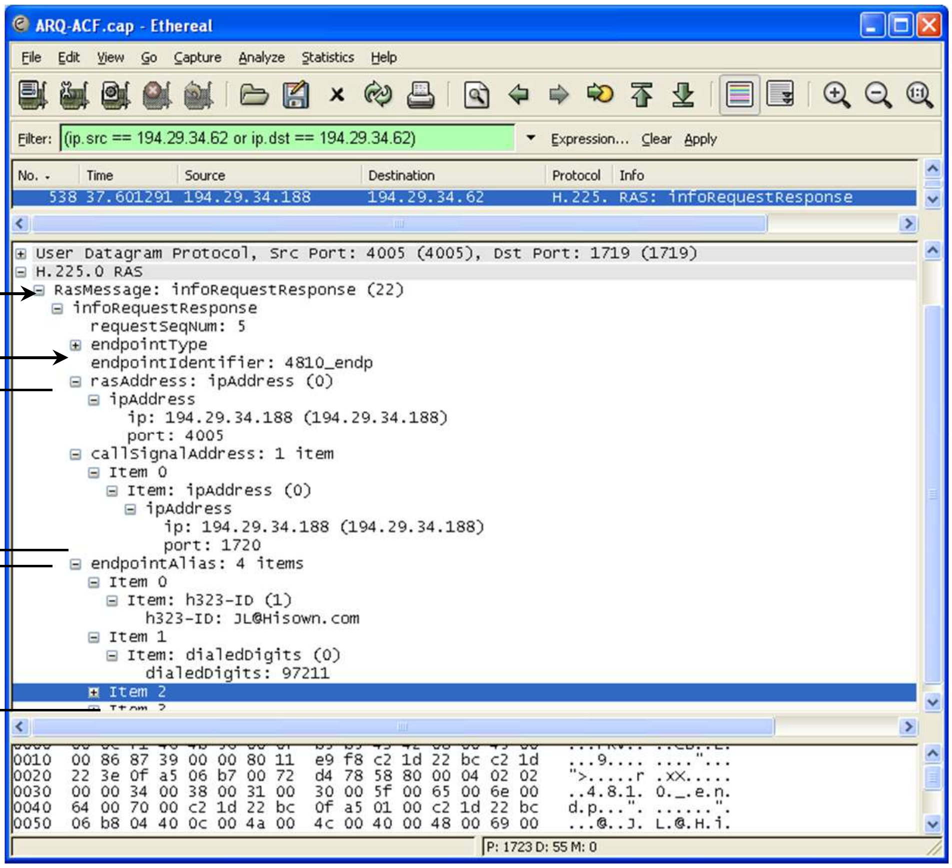
The endpoint will reestablish its connection parameters, as is shown in the capture, below:

Message Type 22 - IRR

Endpoint identifies itself

Endpoint reestablishes its RAS and Call Signal (Q.931) addresses, ensuring GK it is still available.

"Aliases" for this endpoint, such as the logged on user name and the E.164 phone number






(C) 2005,2012 Technologeeks.com - All Rights Reserved

# H.225.0 - RAS

**Bandwidth**

**Bandwidth ReQuest (BRQ)** – Request to gatekeeper for bandwidth increase/decrease

**Bandwidth ConFirm (BCF)** – Acknowledgement indicating requested change has been accepted

**Bandwidth ReJect (BRJ)** – Negative Acknowledgement indicating change has been received, but denied

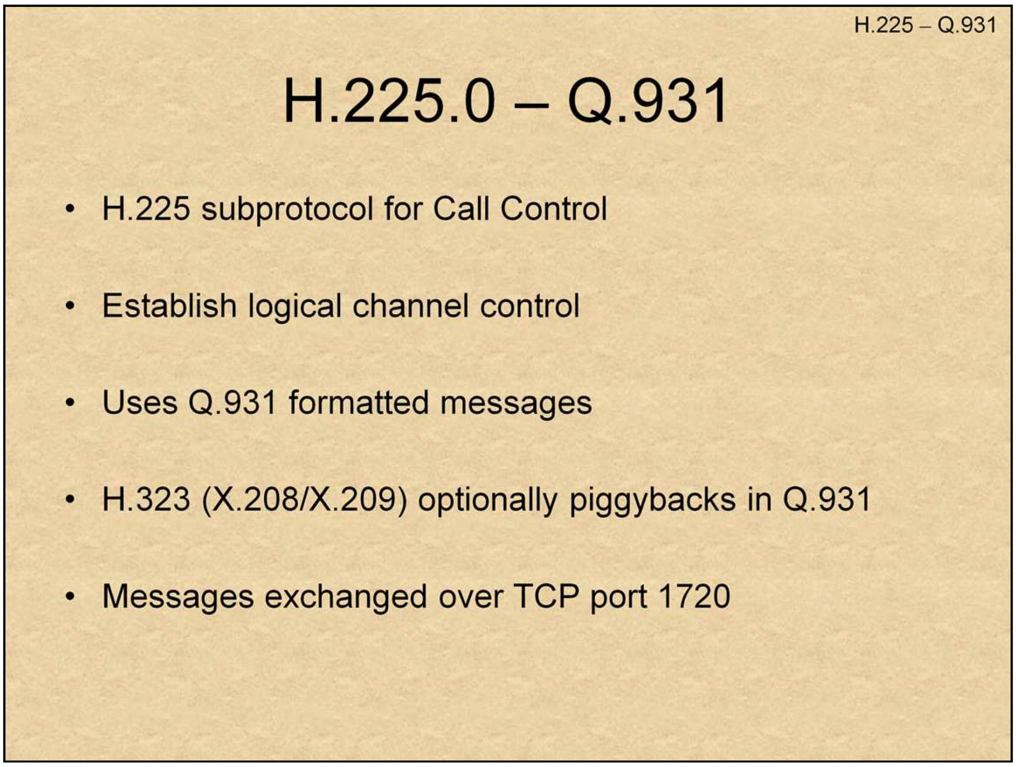
---

**Resource Availability Indicator (RAI)** - used by gateways to inform gatekeeper of current status of resources

**Resource Availability Confirmation (RAC)** – Acknowledgement by Gatekeer on RAI message

The Bandwidth increase/decrease requests are passed between endpoint and gatekeeper, and offer no guarantee as to bandwidth allocated by the rest of the network (i.e. from Gatekeeper and onwards).

(C) 2005,2012 Technologeeks.com - All Rights Reserved



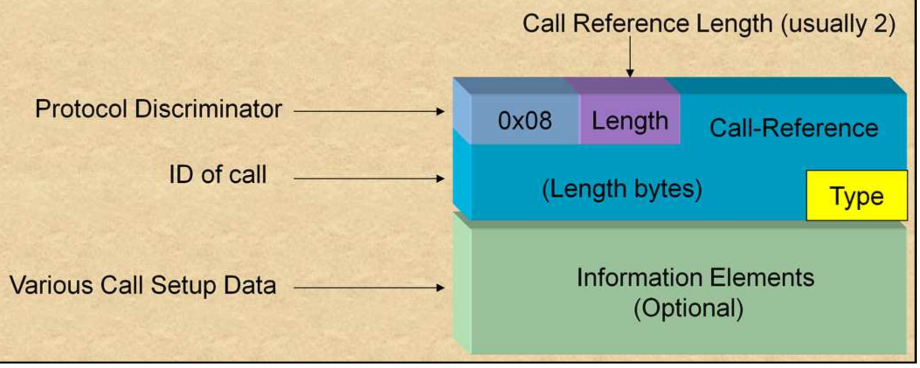

H.225.0/Q.931 is a TOTALLY different protocol, and has little to do with the other component, RAS. Q.931 has been ported from ISDN signaling.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.225 – Q.931

# H.225.0 – Q.931

| Q.931 Message | Type | Purpose |
|---|---|---|
| Setup | 0x5 | “Ring” request |
| Call Proceeding | 0x2 | “Ring” request processed |
| Progress | 0x3 | Progress notification |
| Alerting | 0x1 | “Ringing” |
| Connect | 0x7 | “Pick Up” |




This slide illustrates the Q.931 packet format, and various messages.

Q.931 originated as an ISDN control protocol, and has been adapted to IP by using a subset of its features, that are applicable to Internet settings as well.

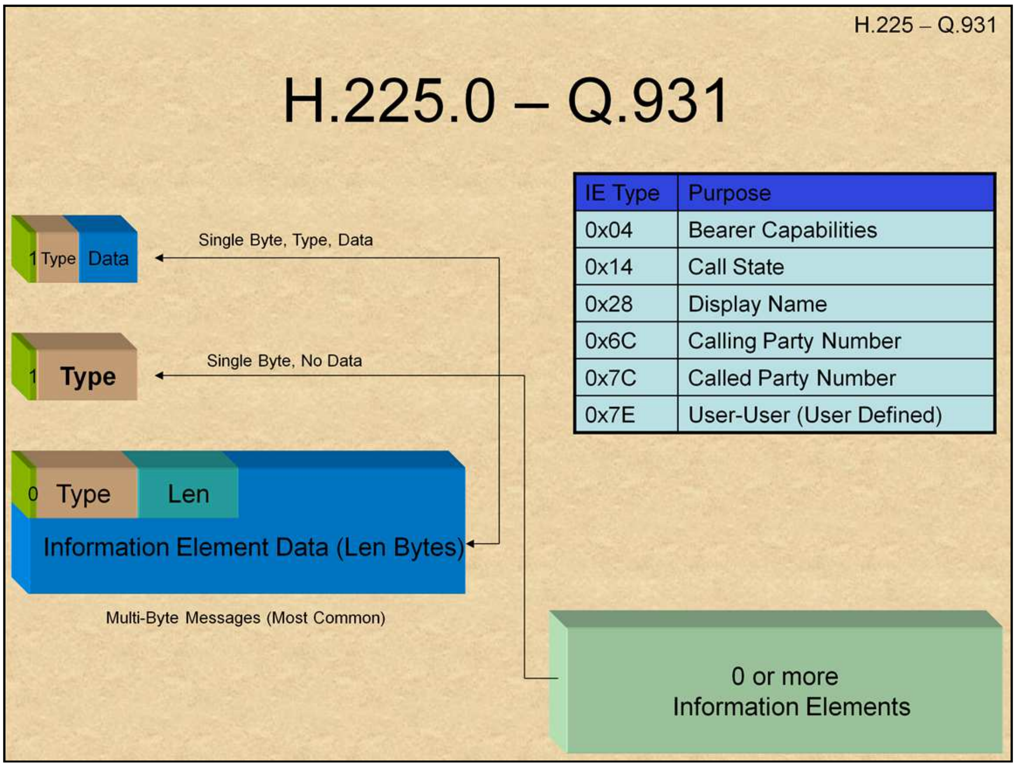
The protocol packets always begin with a fixed byte - “0x08” – this is known as the ***Protocol Discriminator***, and allows detection of Q.931 messages. Following that is the Call-Reference field, applying this Q.931 to a specific call. The call reference field is variable (1-15 bytes), and as such is preceded by a Length field of one byte. Usually, in H.323 settings, it spans two bytes. Following that is the message Type. Common message types are displayed in the table above.

Q.931 messages are actually separated into sub fields: The top 3 bits show message type:

- 000: Call Establishment
- 001: Call State
- 010: Call Clearing (Teardown)
- 011: Miscellaneous.

Optionally, zero or more “Information elements” may be passed along in the header. More on that in the next slide.

(C) 2005,2012 Technologeeks.com - All Rights Reserved



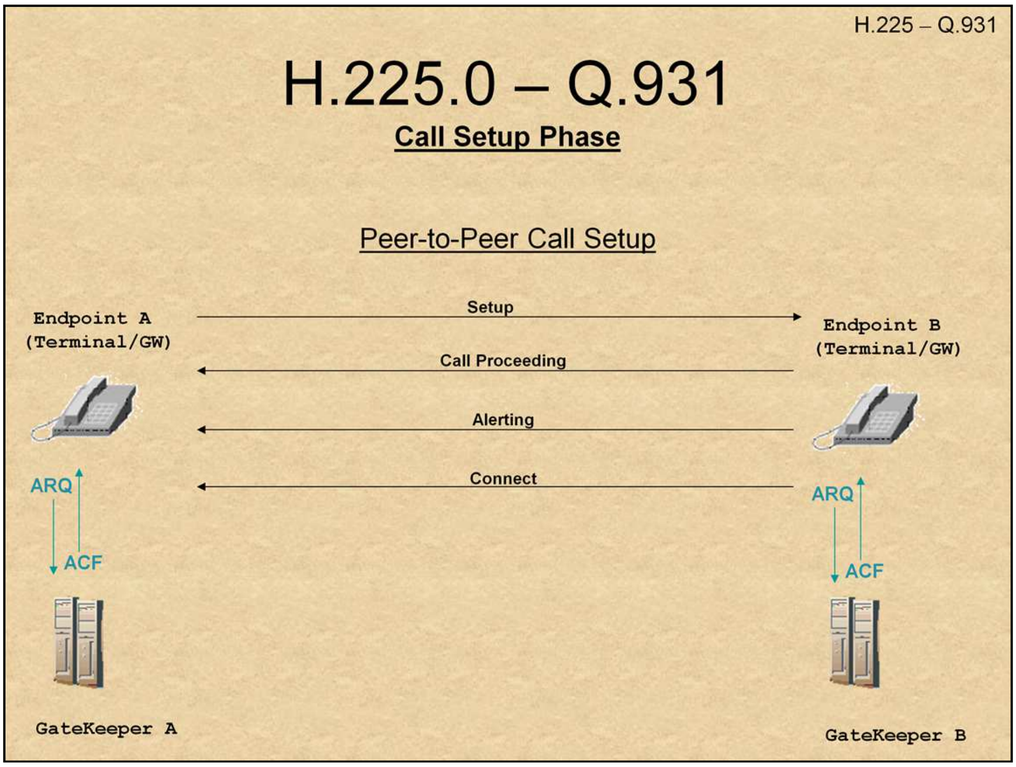

Information Elements may be formatted in one of two types:

Single Byte – If the first bit is '1'. The next three bits are the type, and the remaining four are the payload (data). Optionally, the type may span all 7 bits, if no data is applicable for that type.

Multi-Byte – If the first bit is 0. The next seven bits are then interpreted as the type. The next byte will be the variable Length of the Information Element, and following it will be 'Len' bytes containing the data.

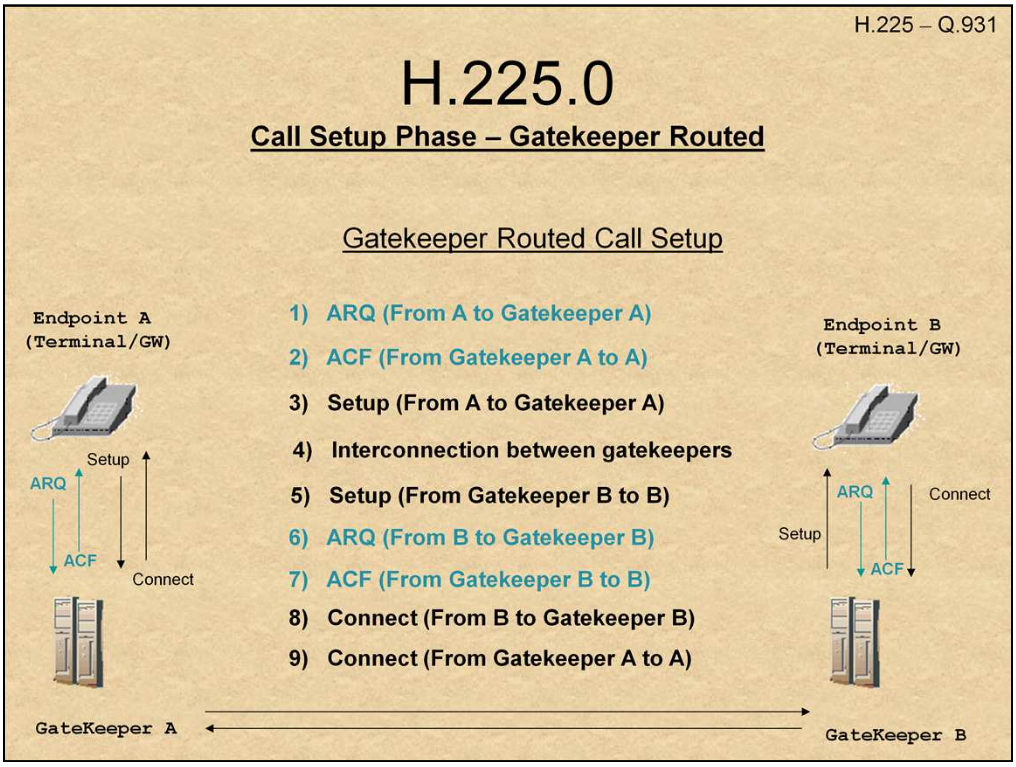
Some IE Types are given in the table, above. 0x7E is used for User Defined messages, in this case H.323 identification messages, that are piggybacked in X.228 or X.229 format.

(C) 2005,2012 Technologeeks.com - All Rights Reserved



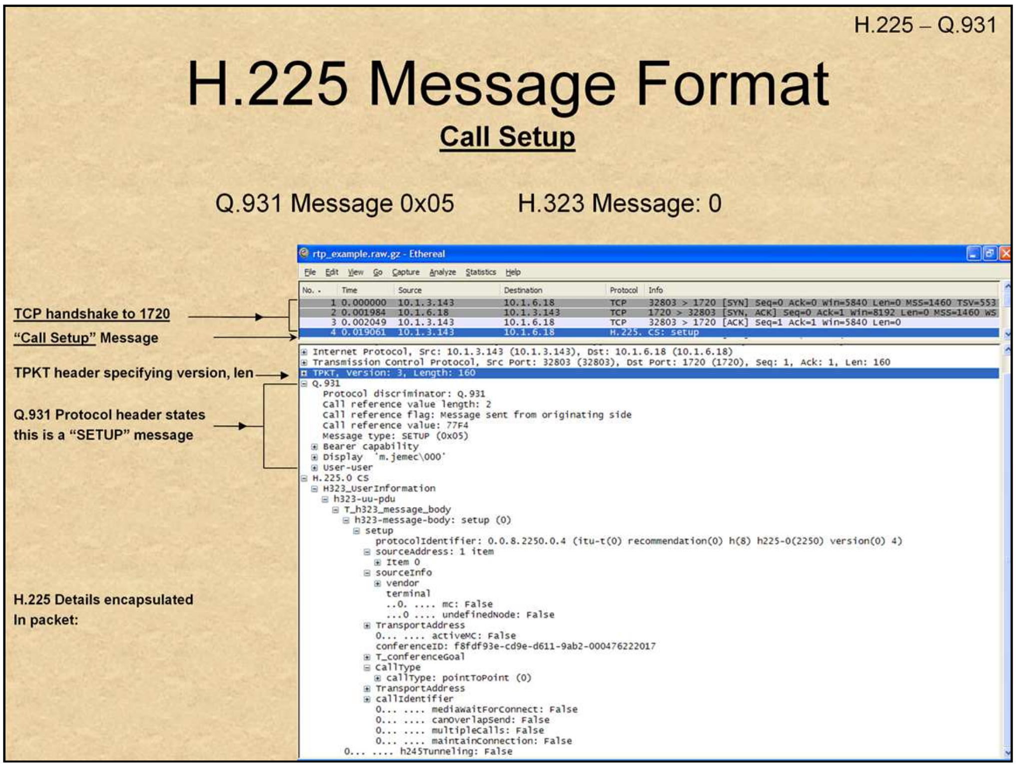

Note that the H.225 messages are not independent of the RAS messages. Both the source and target Gatekeepers (if present) must approve (i.e. send ACFs) the call, otherwise it simply will not happen. For zone-internal calls, the local Gatekeeper is consulted.

(C) 2005,2012 Technologeeks.com - All Rights Reserved




Gatekeeper routed configurations, as the name suggests, enable all calls to be routed through the gatekeepers themselves. Connection is not Point to Point, but Point-GateKeeper-GateKeeper-Point. This is is useful to prevent cases wherein a rogue terminal can attempt connections without GateKeeper approval. In these configurations, the GateKeepers establish the call, and may configure FireWalls to enable specific connections, giving better control over VoIP calls.

(C) 2005,2012 Technologeeks.com - All Rights Reserved





(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.225 – Q.931

# H.225 Message Format

## Call Proceeding

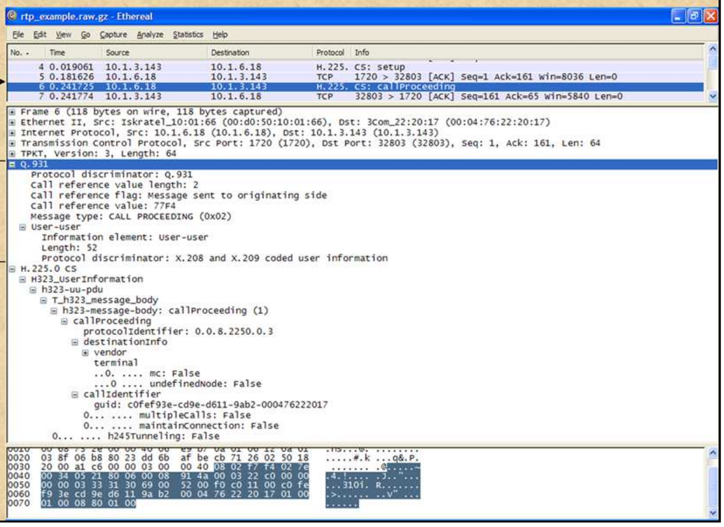
Q.931 Message 0x02 H.323 Message: 1

"Call Proceding" Message

TPKT header specifying version, len

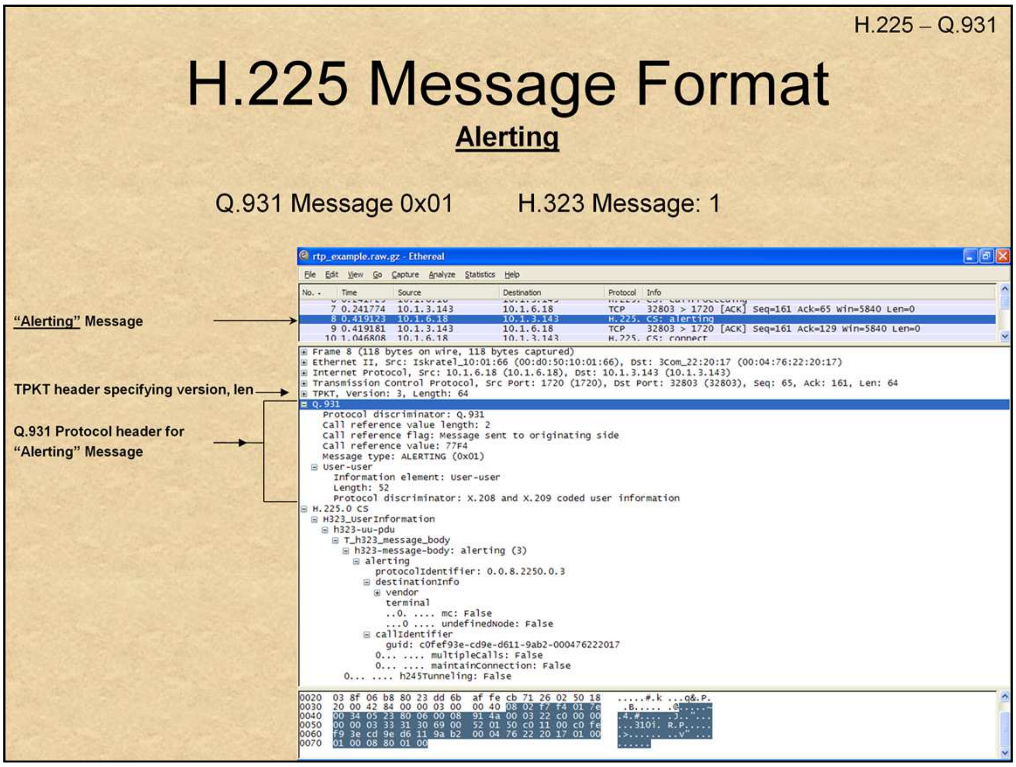
Q.931 Protocol header for "Call Proceeding" Message

H.225 Details encapsulated In packet:






(C) 2005,2012 Technologeeks.com - All Rights Reserved


H.225 – Q.931
H.225 Message Format
Alerting
Q.931 Message 0x01
H.323 Message: 1
"Alerting" Message
TPKT header specifying version, len
Q.931 Protocol header for "Alerting" Message
rtp_example.raw.gz - Ethereal
File Edit View Go Capture Analyze Statistics Help
No. Time Source Destination Protocol Info
7 0.241774 10.1.3.143 10.1.6.18 TCP 32803 > 1720 [ACK] Seq=161 Ack=65 Win=5840 Len=0
8 0.419123 10.1.6.18 10.1.3.143 H.225. CS: alerting
9 0.419181 10.1.3.143 10.1.6.18 TCP 32803 > 1720 [ACK] Seq=161 Ack=129 Win=5840 Len=0
Frame 8 (118 bytes on wire, 118 bytes captured)
Ethernet II, Src: Iskratel_10:01:66 (00:d0:50:10:01:66), Dst: 3Com_22:20:17 (00:04:76:22:20:17)
Internet Protocol, Src: 10.1.6.18 (10.1.6.18), Dst: 10.1.3.143 (10.1.3.143)
Transmission Control Protocol, Src Port: 1720 (1720), Dst Port: 32803 (32803), Seq: 65, Ack: 161, Len: 64
TPKT, Version: 3, Length: 64
Q.931
Protocol discriminator: Q.931
Call reference value length: 2
Call reference flag: Message sent to originating side
Call reference value: 77F4
Message type: ALERTING (0x01)
User-user
Information element: User-user
Length: 52
Protocol discriminator: X.208 and X.209 coded user information
H.225.0 CS
H323_UserInformation
h323-uu-pdu
T_h323_message_body
h323-message-body: alerting (3)
alerting
protocolIdentifier: 0.0.8.2250.0.3
destinationInfo
vendor
terminal
..0. .... mc: False
...0 .... undefinedNode: False
callIdentifier
guid: c0fef93e-cd9e-d611-9ab2-000476222017
0... .... multipleCalls: False
0... .... maintainConnection: False
0... .... h245Tunneling: False
0020 03 8f 06 b8 80 23 dd 6b af fe cb 71 26 02 50 18
0030 20 00 42 84 00 00 03 00 00 40 08 02 f7 f4 01 7e
0040 00 34 05 23 80 06 00 08 91 4a 00 03 22 c0 00 00
0050 00 00 03 33 31 30 69 00 52 01 50 c0 11 00 c0 fe
0060 f9 3e cd 9e d6 11 9a b2 00 04 76 22 20 17 01 00
0070 01 00 08 80 01 00

H.225 – Q.931

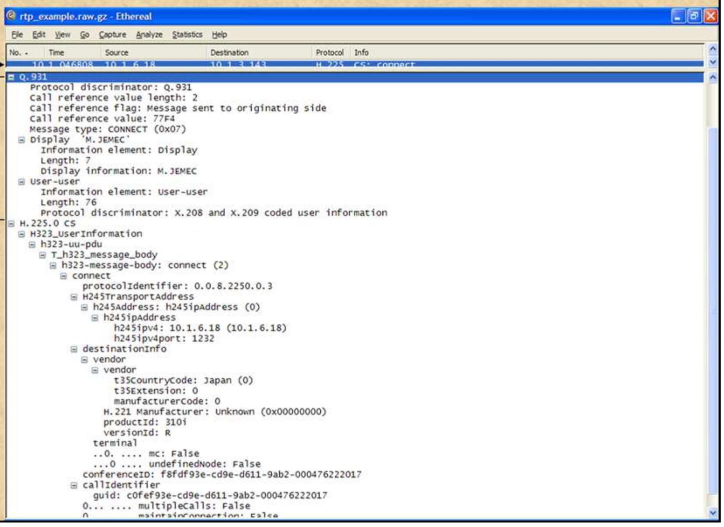
# H.225 Message Format

**Connect**

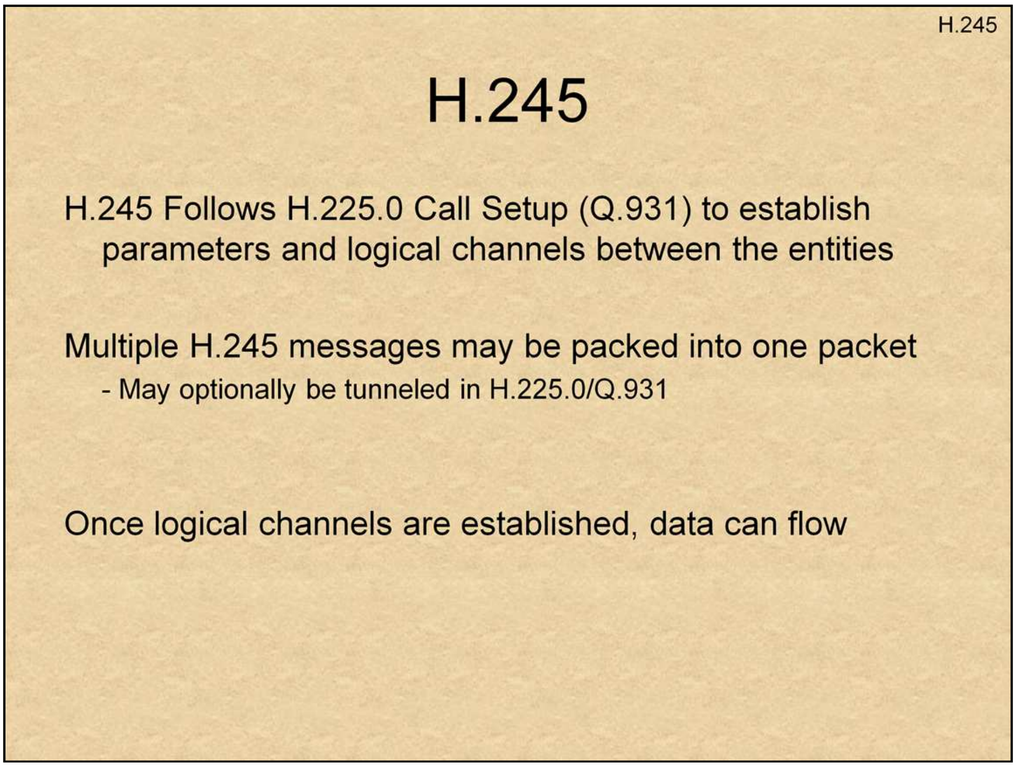
Q.931 Message 0x07 H.323 Message: 2

**"Connect" Message**

**Q.931 Protocol header for "Connect" Message**




(C) 2005,2012 Technologeeks.com - All Rights Reserved




Once H225.0 sets up a logical call "object", another protocol comes into play – this time, H.245. This protocol is used in call control, to set call parameters and establish logical channels between the peers.

H.245 encapsulation, or "tunneling" is quite useful, as it:

- Avoids using an ephemeral port for H.245 (making it firewall friendly)
- Achieves greater synchronization between the call signaling channel and call control channels.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.245

# H.245.0

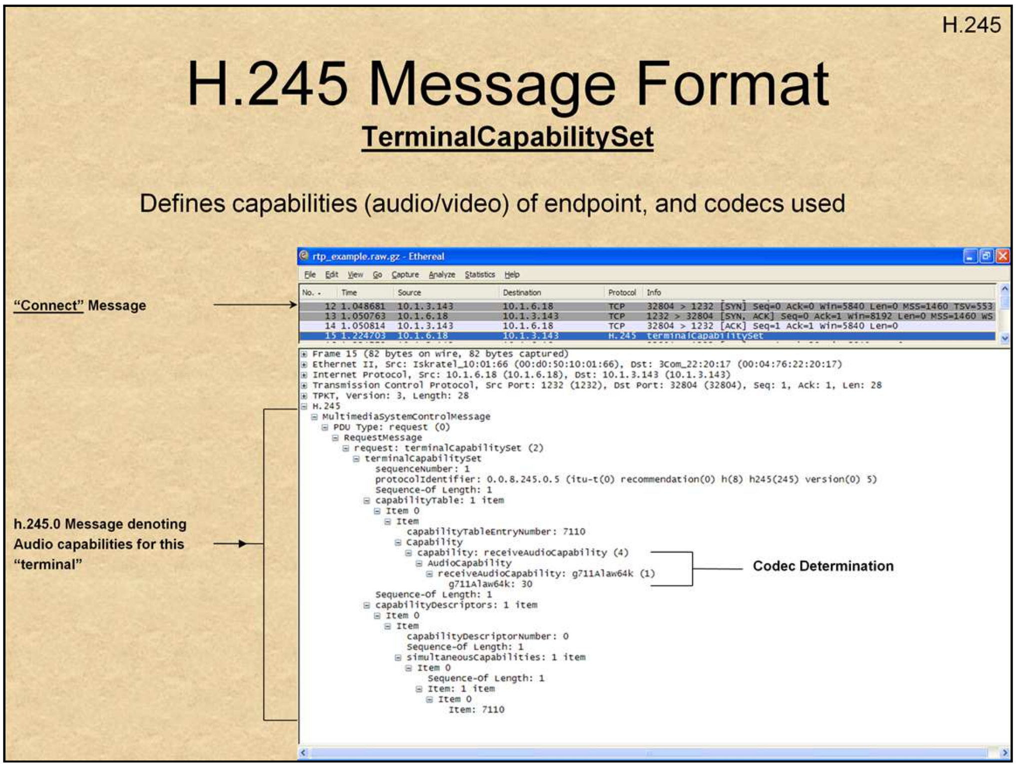
## Logical Channel Setup

Messages are exchanged over ephemeral ports


| H.245 Message | Purpose |
|---|---|
| TerminalCapabilitySet[Ack] | Set endpoint capabilities and codecs |
| MasterSlaveDeterminiation[Ack] | Define role of endpoint in call |
| OpenLogicalChannel[Ack] | Setup transport for call |

Note that the H.225 messages are not independent of the RAS messages. Both the source and target Gatekeepers (if present) must approve (i.e. send ACFs) the call, otherwise it simply will not happen. For zone-internal calls, the local Gatekeeper is consulted.

(C) 2005,2012 Technologeeks.com - All Rights Reserved





(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.245

# H.245 Message Format

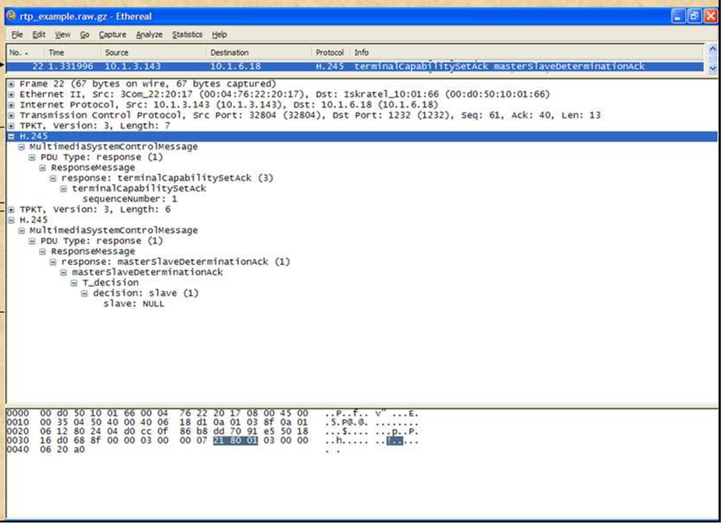
## Acknowledgement Messages

Acknowledgement messages are marked as Response PDUs (1)

Acknowledgement" Messages

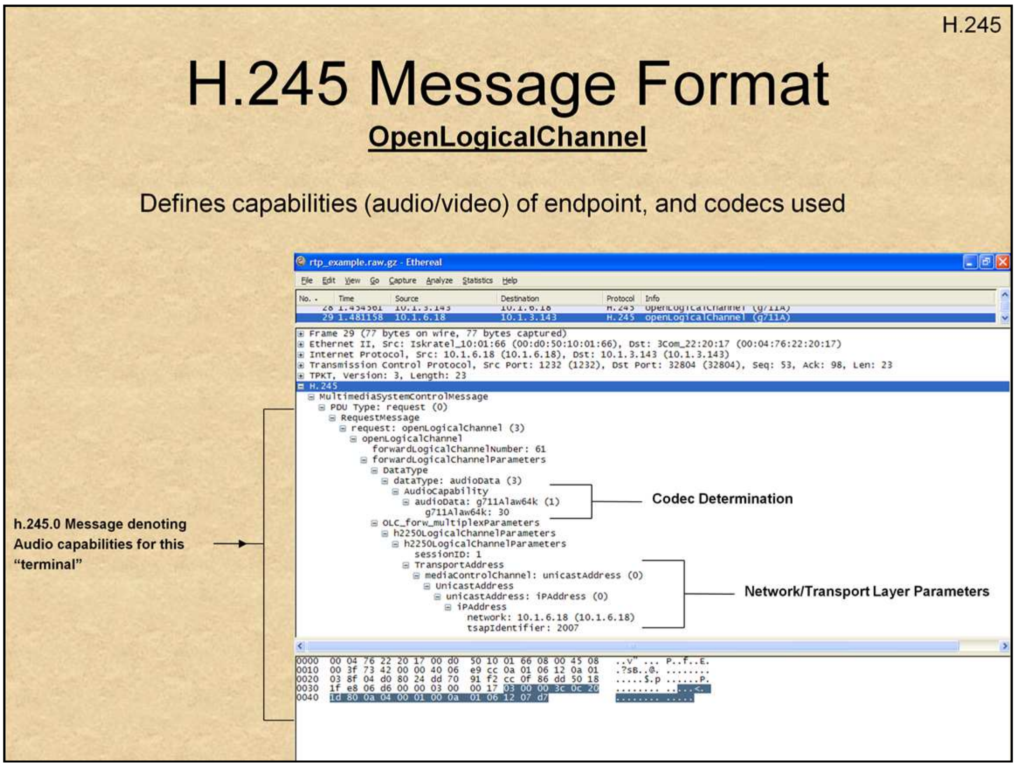
h.245.0 Message denoting acknowledgement of peer capabilities

h.245.0 Message denoting this endpoint will assume the role of a slave






(C) 2005,2012 Technologeeks.com - All Rights Reserved


H.245
H.245 Message Format
OpenLogicalChannel
Defines capabilities (audio/video) of endpoint, and codecs used
rtp_example.raw.gz - Ethereal
File Edit View Go Capture Analyze Statistics Help
No. Time Source Destination Protocol Info
29 1.481158 10.1.6.18 10.1.3.143 H.245 openLogicalChannel (g711A)
Frame 29 (77 bytes on wire, 77 bytes captured)
Ethernet II, Src: Iskratel_10:01:66 (00:d0:50:10:01:66), Dst: 3Com_22:20:17 (00:04:76:22:20:17)
Internet Protocol, Src: 10.1.6.18 (10.1.6.18), Dst: 10.1.3.143 (10.1.3.143)
Transmission Control Protocol, Src Port: 1232 (1232), Dst Port: 32804 (32804), Seq: 53, Ack: 98, Len: 23
TPKT, Version: 3, Length: 23
H.245
MultimediaSystemControlMessage
PDU Type: request (0)
RequestMessage
request: openLogicalChannel (3)
openLogicalChannel
forwardLogicalChannelNumber: 61
forwardLogicalChannelParameters
DataType
dataType: audioData (3)
AudioCapability
audioData: g711Alaw64k (1)
g711Alaw64k: 30
OLC_forw_multiplexParameters
h2250LogicalChannelParameters
h2250LogicalChannelParameters
sessionID: 1
TransportAddress
mediaControlChannel: unicastAddress (0)
UnicastAddress
unicastAddress: iPAddress (0)
iPAddress
network: 10.1.6.18 (10.1.6.18)
tsapIdentifier: 2007
Codec Determination
Network/Transport Layer Parameters
h.245.0 Message denoting Audio capabilities for this “terminal”
0000 00 04 76 22 20 17 00 d0 50 10 01 66 08 00 45 08
0010 00 3f 73 42 00 00 40 06 e9 cc 0a 01 06 12 0a 01
0020 03 8f 04 d0 80 24 dd 70 91 f2 cc 0f 86 dd 50 18
0030 1f e8 06 d6 00 00 03 00 00 17 03 00 00 3c 0c 20
0040 1d 80 0a 04 00 01 00 0a 01 06 12 07 d7

H.245

# H.225/H.245 “Fast Start”

H.245 Messages may be tunneled in H.225/Q.931

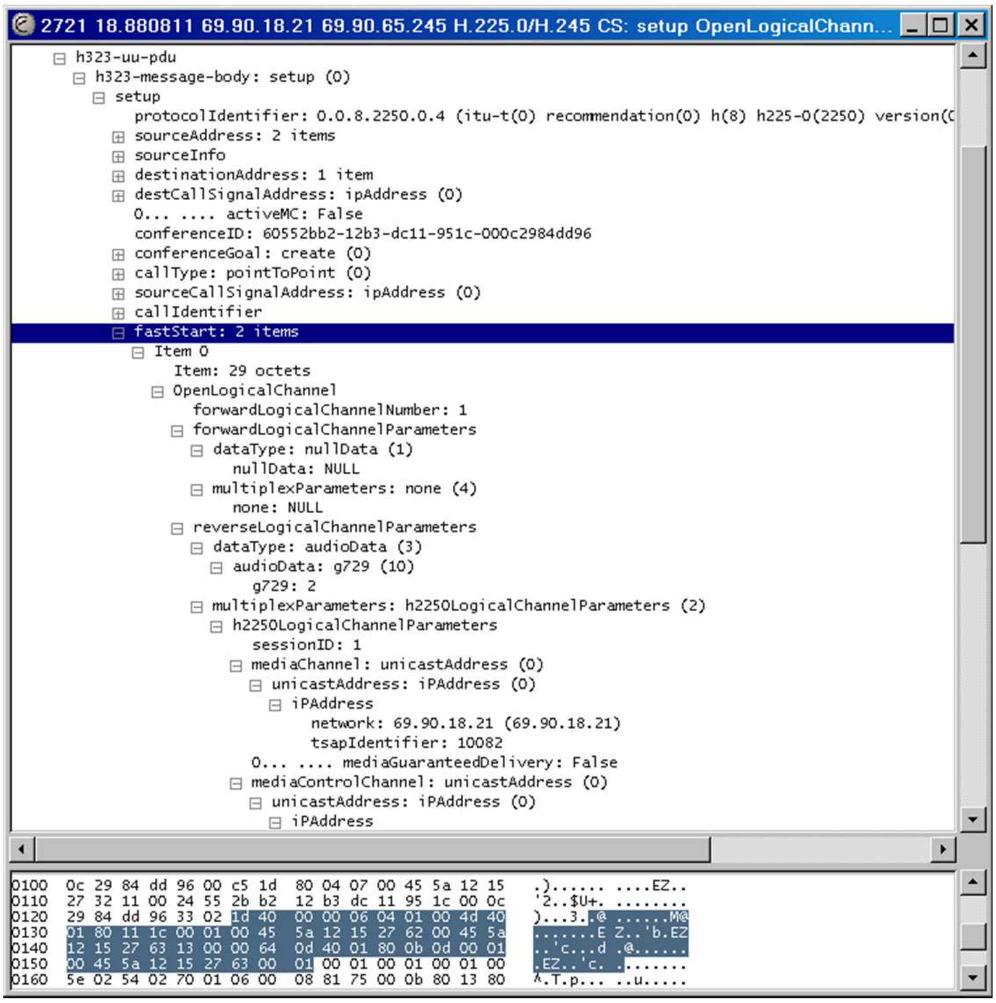
- OLC/TCS etc. sent in SETUP, ALERTING, PROGRESS or CONNECT

This can be performed even before the handshake is done

Commonly used, as of H.323v2:

- Allows for more efficient communication with less sockets
- Significantly lowers the initial call overhead.
    - Voice/Video flow begins immediately with no delay

H.323v2 and above support a special form of H.245 tunneling called “Fast Connect” or “Fast Start”. This is the sending of optional H.245 “OpenLogicalChannel” messages, that open the RTP media stream, in the SETUP and ALERTING messages. This is meant to prevent the H.245 setup overhead – which often results in voice clipping or silence. The messages are sent in the “fastStart” Information Element of the Q.931 messages.



H.235

## So where's security in all this?

Yet another H.xxx protocol – H.235v2 (Annex D)

Provides:
- Password based security (Baseline security)
- Digital Certificate Support (Signature Security)

Security "Features" are obsolete by today's standards.
VERY poorly documented.

Sub-standards define additional profiles

So where's Security? Good question.

H.235 devines in its Annex D the "minimal set of requirements" for security. However, as it turns out, this really IS the bare minimum. "Baseline Security" = password based, shared key. Not overly scalable or practical.

Additional profiles are further defined:

- **235.1** Baseline security profile
- **235.2** Signature security profile
- **235.3** Hybrid security profile
- **235.4** Direct and selective routed call security
- **235.5** Framework for secure authentication in RAS using weak shared secrets
- **235.6** Voice encryption profile with native H.235/H.245 key management
- **235.7** Usage of the MIKEY key management protocol for the Secure RTP
- **235.8** Key exchange for SRTP using secure signaling channels
- **235.9** Security gateway support for H.323

(C) 2005,2012 Technologeeks.com - All Rights Reserved

# H.235v2

Authentication: HMAC-SHA1-96
Integrity: HMAC-SHA1-96

Key Management:
- H.225.0 – Diffie Hellman
- H.245 – Shared Key (by DH) for DES/3-DES

Directory Services implemented by H.350v2

RTP payload is DES/3DES or weak RC2 (optionally AES)

It's important to emphasize that H.235 handles authentication/encryption.
DoS issues are not handled in any way.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

# H.323 and Firewalls

## H.323v6 adds recommendations for NATs and Firewalls:

| Recommendation | Ideas |
|---|---|
| H.460.17<br>("RAS over H.225") | - RAS messages tunneled in Q.931 FACILITY messages<br>- Endpoint on internal network contacts external FW<br>- Same channel used for RAS, H.225 ***and*** H.245<br>- Detected by a DNS SRV record (._tcp.) |
| H.460.18 | - "Knock-Knock" approach:<br>- External→Internal: RAS SCI message ("Knock knock")<br>- Internal→External: RAS SCR reply ("Commmmming")<br>- Internal→External: Q.931 FACILITY (Opening the door)<br>- External→Internal: SETUP<br>- Slight H.245 modifications |
| H.460.19 | RTP/RTCP keep-alive streams (from recipient to sender) |

H.323v6 has incorporated the recommendations in H.460.17, H.460.18, and H.460.19.

The former two (.17,.18) are used to resolve issues introduced by FireWalls in the signaling channel. The latter (.19) is used to prevent the media channel from disconnecting

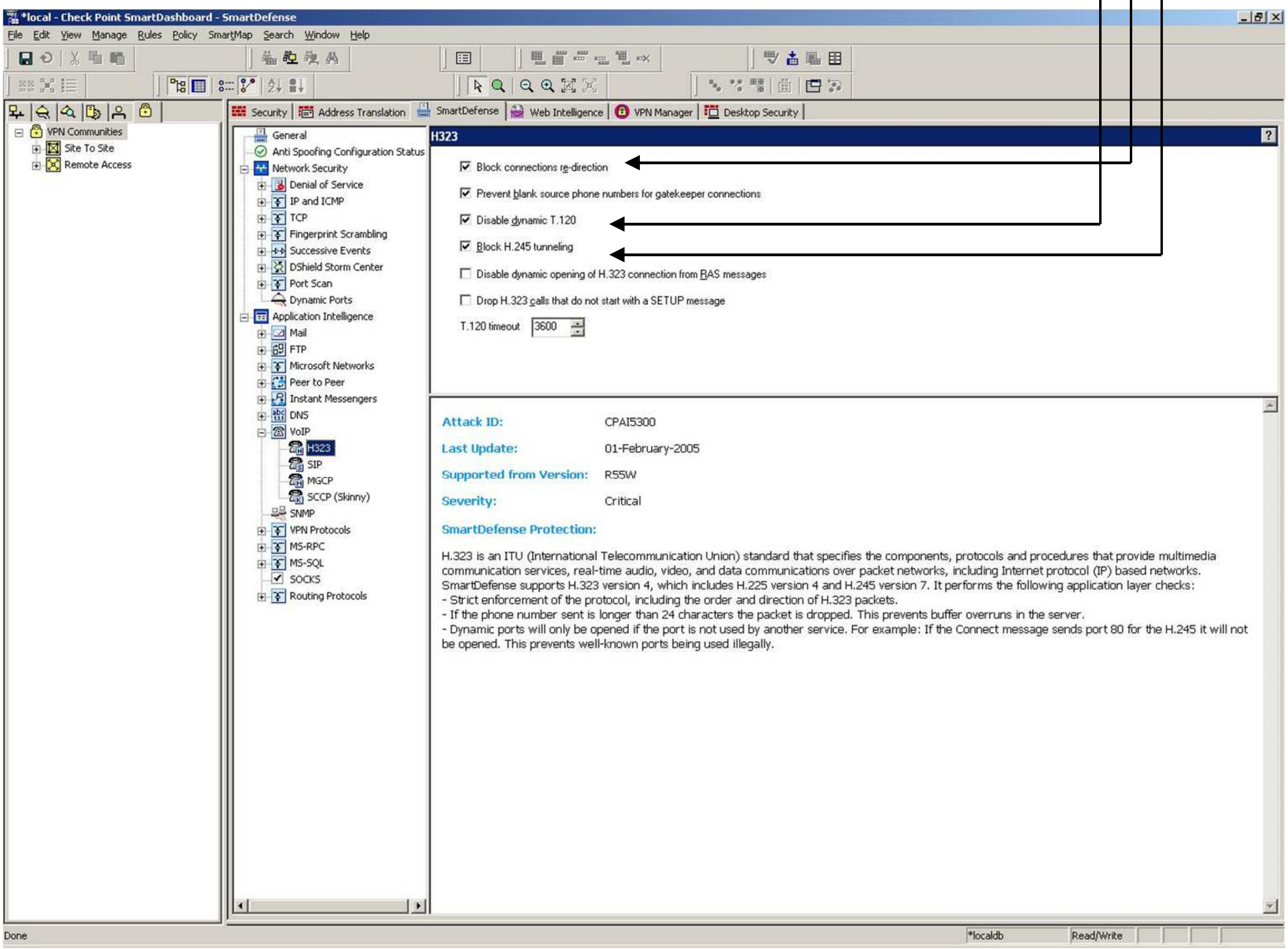
http://www.h323forum.org/papers/301005_Firewall_NAT_Traversal_White_Paper.pdf, a presentation by RadVision, serves as a good reference for these features.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

H.323

# H.323 & Firewall-1

- Force/Block H.245 in Q.931
- Force all calls to be setup through GateKeeper
- Validate Phone IDs and restrict fields
- Enforce Q.931 flow (“setup”)
- Disable ephmeral ports (e.g. T.120)





(C) 2005,2012 Technologeeks.com - All Rights Reserved

# SCCP

Selsius Call Control Protocol (a.k.a “Skinny”)

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SCCP

# SCCP (“Skinny”)

- Cisco Proprietary protocol, also Q.713
- Used in IP Phone ←→Call Mgr. communications
- Utilizes TCP port 2000
- Protocol is lightweight and minimal
- Phone is really a “dumb terminal” controlled by CCM

The Cisco “Skinny” protocol was originally developed by the Selsius Corporation. With their acquisition by Cisco, this became a Cisco propietary protocol, that is used in the communication between the Cisco IP Phones (mostly 79xx) and the Cisco Call Manager.

The protocol is a very lightweight one (hence the nickname “Skinny”). The Call Manager does all the H.323 and SIP processing, acting as a proxy, leaving the IP Phone the task of processing the VoIP RTP datastream.

The protocol is rather scarcely documented, as full documentation is available only to Cisco affiliates. The rest of this section attempts to explain this protocol, thanks to a lot of research, packet captures, and common sense.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SCCP

# SCCP ("Skinny") Messages

(in order of appearance)

**Stage I** – Phone/CallMgr registration

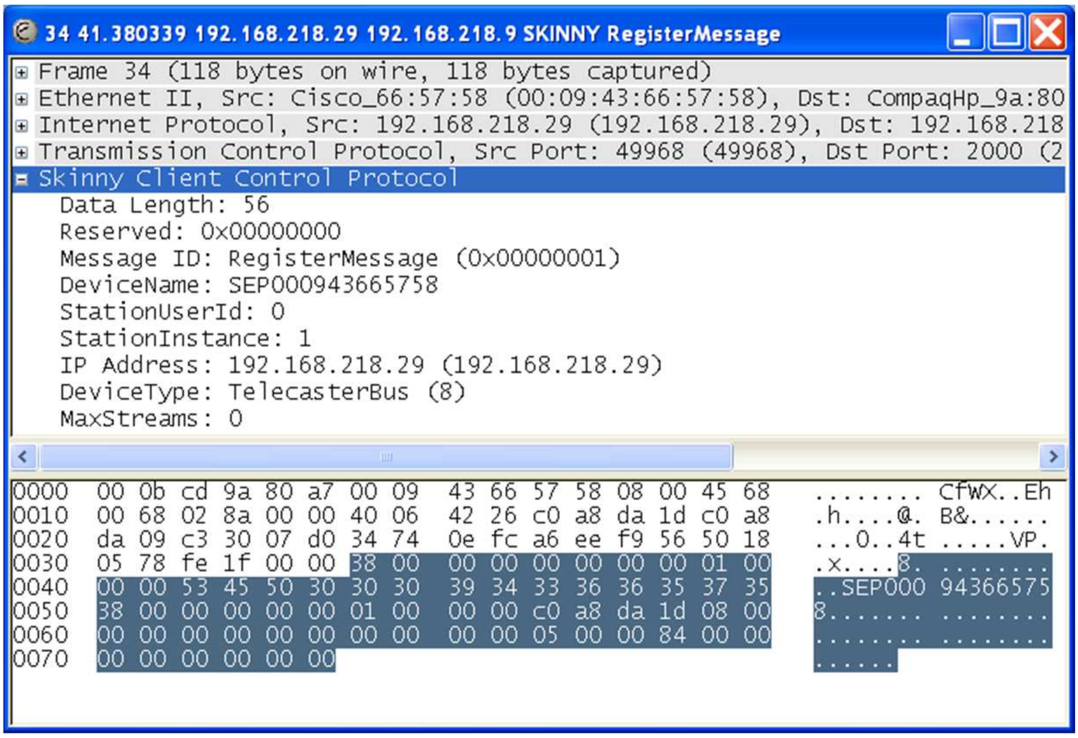
| | Msg | Usage | Data |
|---|---|---|---|
| → | 0001 | RegisterMessage | Device Name, Station UserID & Instance, IP Address, Device Type, Max Streams |
| → | 0002 | IPPortMessage | IP and Port Terminal is listening on |
| ← | 0081 | RegisterAckMessage | Keep Alive Interval, Date Template (M/D/YA), Secondary Keep Alive Interval |
| ← | 009B | CapabilitiesRequest | Call Mgr asks for Station capabilties |
| → | 0010 | CapabilitiesResponse | CapCount capabilities(PayLoad/MaxFramesPerPacket) |
| → | 000F | VersionRequest | Station requests Call Mgr version |
| ← | 0098 | VersionResponse | Call Mgr Version |
| → | 000E | ButtonTemplateRequest | -- |
| ← | 0097 | ButtonTemplateMessage | Button offset/count and 40-something button defs |
| → | 000D | TimeDateRequest | -- |
| ← | 0094 | DefineTimeDate | Y/M/WD/D, Hour/Min/Sec/mSec, 32-bit TimeStamp |

→: Phone to Call Mgr ←: Call Mgr to phone

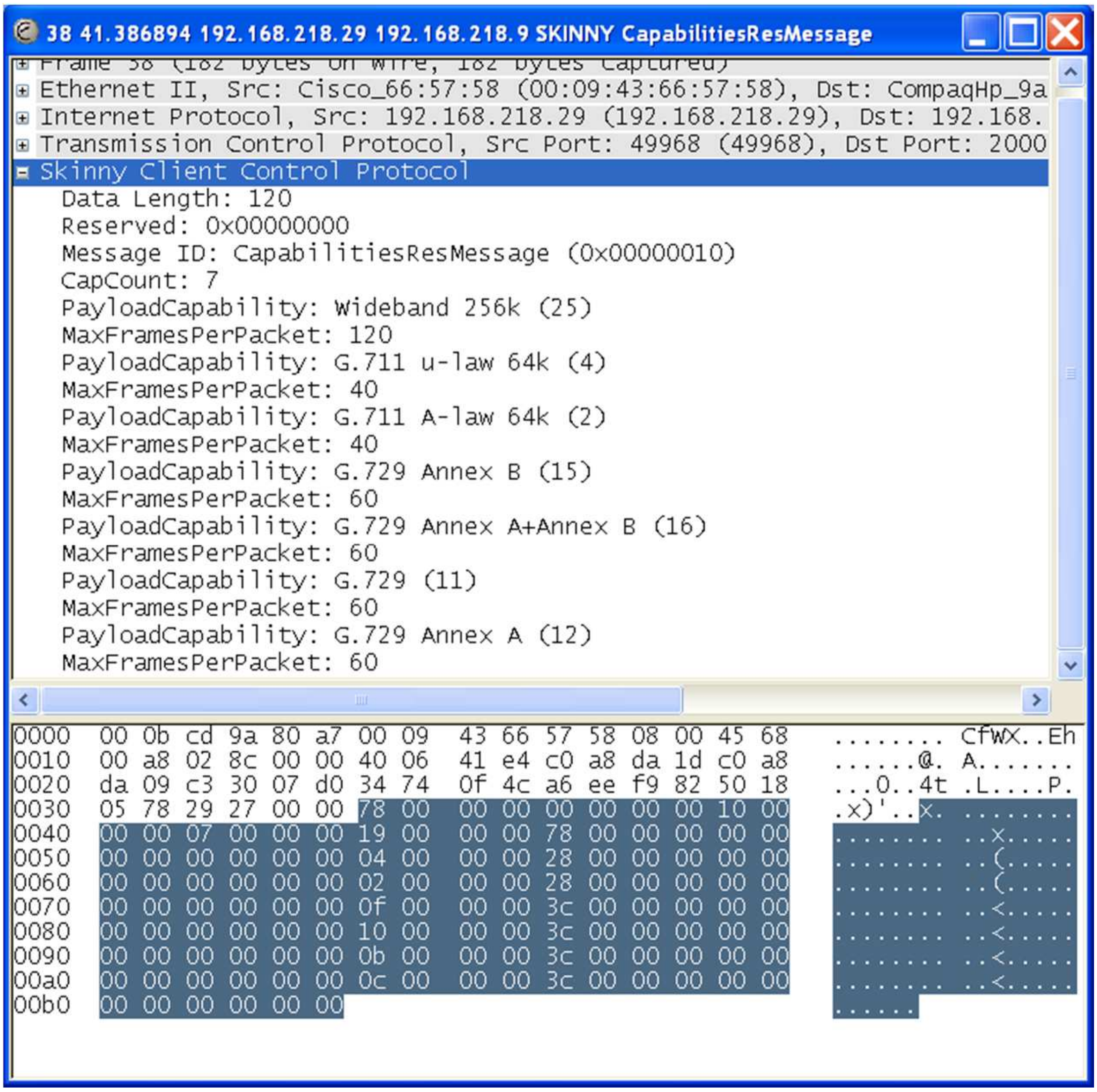
The table above shows the SCCP message type, as they "appear" in the lifespan of a telephone. In particular, this table shows the phone registration process with the call manager.

The phone registers its IP, as well as its type and name. The CCM asks it to provide its "capabilities" (voice/video codecs supported). It then caches the IP-Phone capabilities and translates them to H.323 capabilities.

The illustration to the right depicts a typical Registration message, as captured by Ethereal's protocol dissector.



The Capabilities Response message is shown in the following illustration:




(C) 2005,2012 Technologeeks.com - All Rights Reserved

SCCP

# SCCP ("Skinny") Messages

(in order of appearance)

**Stage I ½** – Keep Alive/Alarm Messages

| | Msg | Usage | Data |
|---|---|---|---|
| → | 0000 | KeepAliveMessage | -- (sent periodically by phone) |
| ← | 0100 | KeepAliveAckMessage | -- (sent periodically by callMgr) |
| → | 0020 | Alarm Message | Alarm Severity, Display Message & Params |

**Stage II** – Picking up the handset

| | Msg | Usage | Data |
|---|---|---|---|
| → | 0006 | OffHookMessage | -- |
| ← | 0099 | DisplayTextMessage | ASCII text, NULL terminated |
| ← | 0086 | SetLampMessage | Stimulus, StimulusInstance, LampMode |
| ← | 0111 | CallStateMessage | Call State (code), Line Instance, Call Ident |
| ← | 0112 | DisplayPromptStatus | Timeout, DisplayMessage*, Line Inst, Call Ident |
| ← | 0110 | SelectSoftKeysMessage | Line Instance, Call Ident, SoftKeySet, SoftKeyMap (16-bit bitmap) |
| ← | 0116 | ActivateCallPlaneMessage | Line Instance |
| ← | 0082 | StartToneMessage | Dial Tone (as 32 bit identifier) |

→: Phone to Call Mgr ←: Call Mgr to phone

The phone periodically sends "KeepAlive" messages to the CCM (as instructed by the CCM during the registration). Alarms are sent in case of errors – network errors, mostly, such as a phone's inability to load a file from the TFTP, etc.

When a user picks up the handset, the phone sends an "OffHook" message to the CCM. The CCM, in turn, tells the phone e-x-a-c-t-l-y what to do. From the lamp on/off, through the prompt, key settings, and even the dialtone.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SCCP

# SCCP ("Skinny") Messages

(in order of appearance)

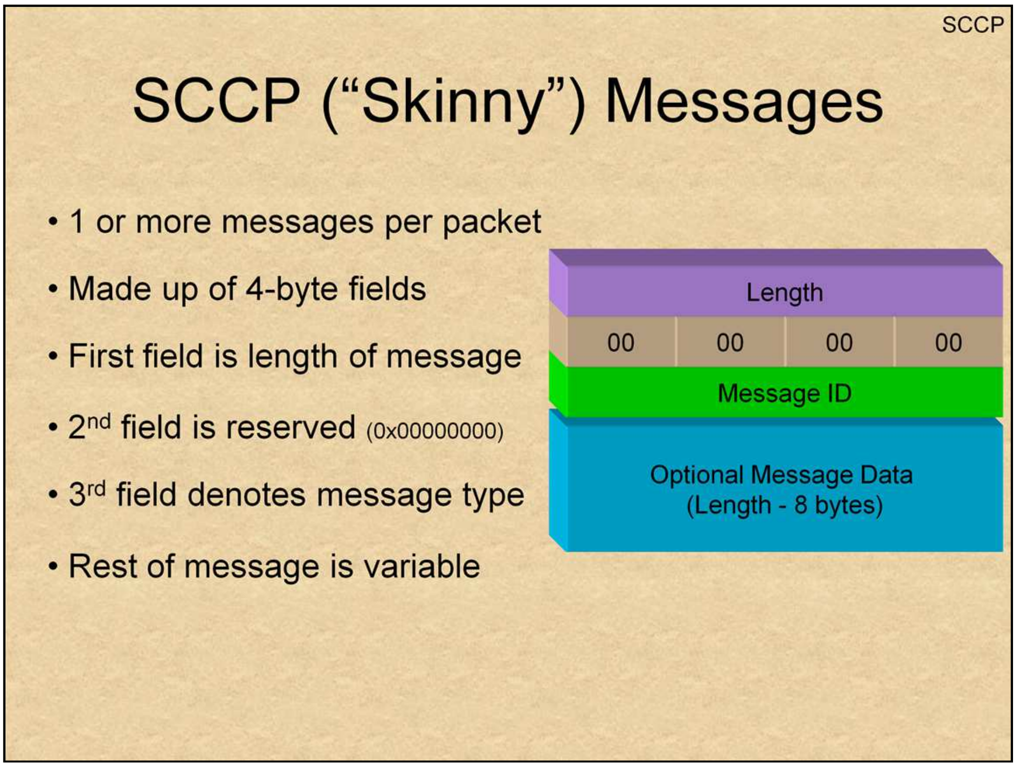
**Stage III** – Placing a call

| | Msg | Usage | Data |
|---|---|---|---|
| → | 0003 | KeyPadButtonMessage | Dialed Digit |
| ← | 0083 | StopToneMessage | 0110 may follow to reconfigure softkeys.. |
| ← | 008F | CallInfoMessage | Calling/Called Party & Party Names, Line Inst., Call Ident, Call Type, Orig. called party |
| ← | 0105 | OpenReceiveChannel | Receive Channel Details.. |
| ← | 008A | StartMediaTransmission | Transmission Channel Details.. |
| → | 0022 | OpenReceiveChannelAck | Status, IP, Port, Pass Through Party ID |
| → | 0007 | OnHookMessage | -- (serves as a call hangup) |
| ← | 0113 | ClearPromptStatusMess.. | Line Instance, Call Ident |
| ← | 0106 | CloseReceiveChannel | Conf Id, Pass Through Party Id |
| ← | 008B | StopMediaTransmission | Conf Id, Pass Through Party Id |

→: Phone to Call Mgr ←: Call Mgr to phone

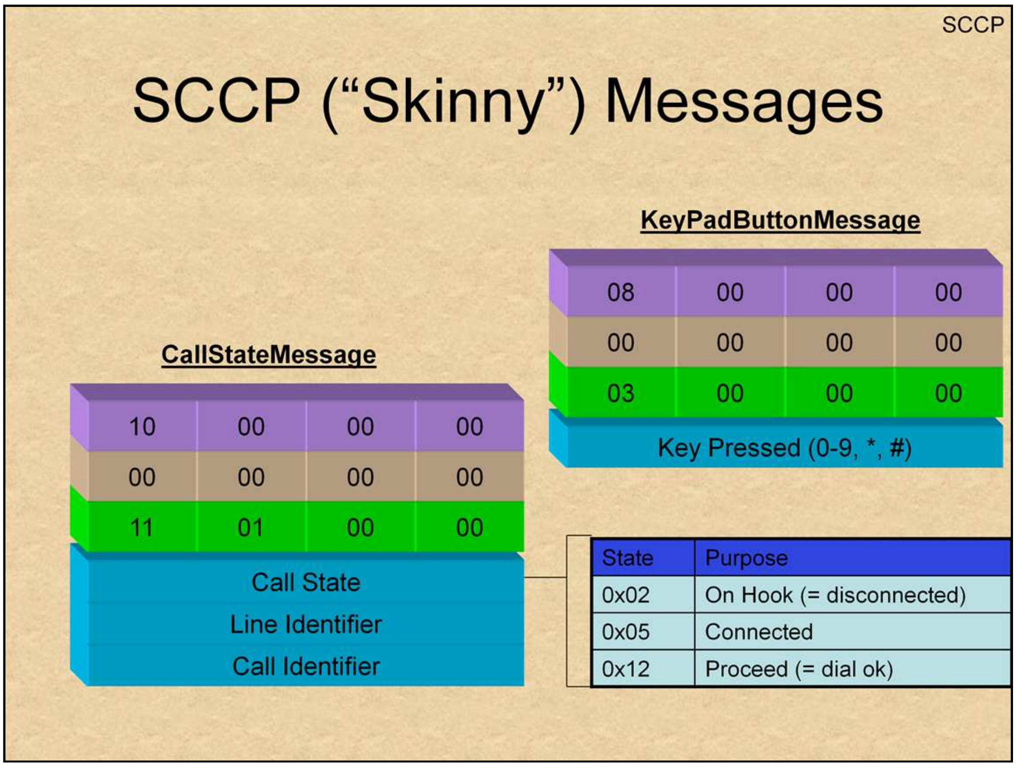
The phone signals the end of a call by an "OnHook" message, telling the call manager the user replaced the handset (therefore hung up the call). It's then that the Call Manager tells the phone to stop transmitting, close the channels, set the call State to OnHook (= disconnected), and present the default user prompt.

(C) 2005,2012 Technologeeks.com - All Rights Reserved




As stated, SCCP is an extremely simple (and wasteful(!)) protocol. The slide above depicts the basic format of a SCCP message. All "fields" are 4 bytes (i.e. words), for easier processing at the phone side. The first field is the length of the message (i.e. the rest of the fields, excluding the "reserved" field, next, which is always zero). Then, the message type – and, if applicable, message arguments. Most messages, however, are of fixed size, as they have a predefined number of arguments. The messages containing strings, however (usually NULL terminated), may differ.

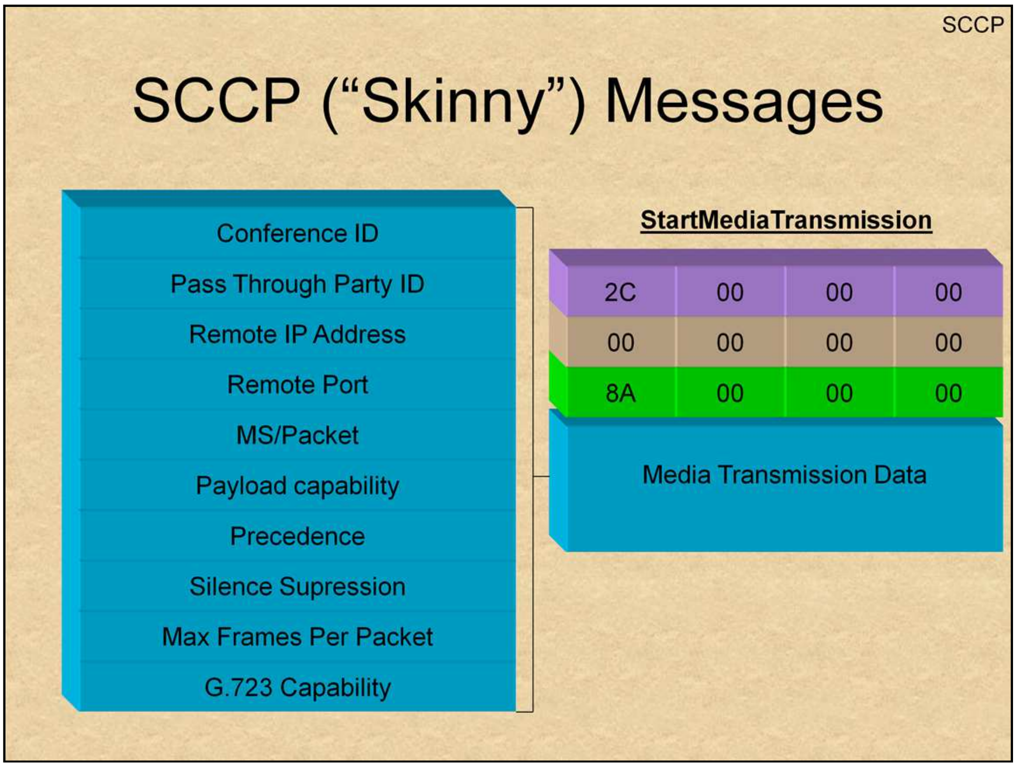
(C) 2005,2012 Technologeeks.com - All Rights Reserved




The slide above shows the important "dialing" messages that SCCP supports. These are the KeyPadButton Message (for each dialed digit) and the CallState Message. The latter is sent by the Call Manager to the Station at various stages of the call lifespan, with the codes specified in the table above.

Note, again, that the protocol is VERY wasteful. Each digit is sent on its own in a KeyPadButton Message (as one byte out of the four).

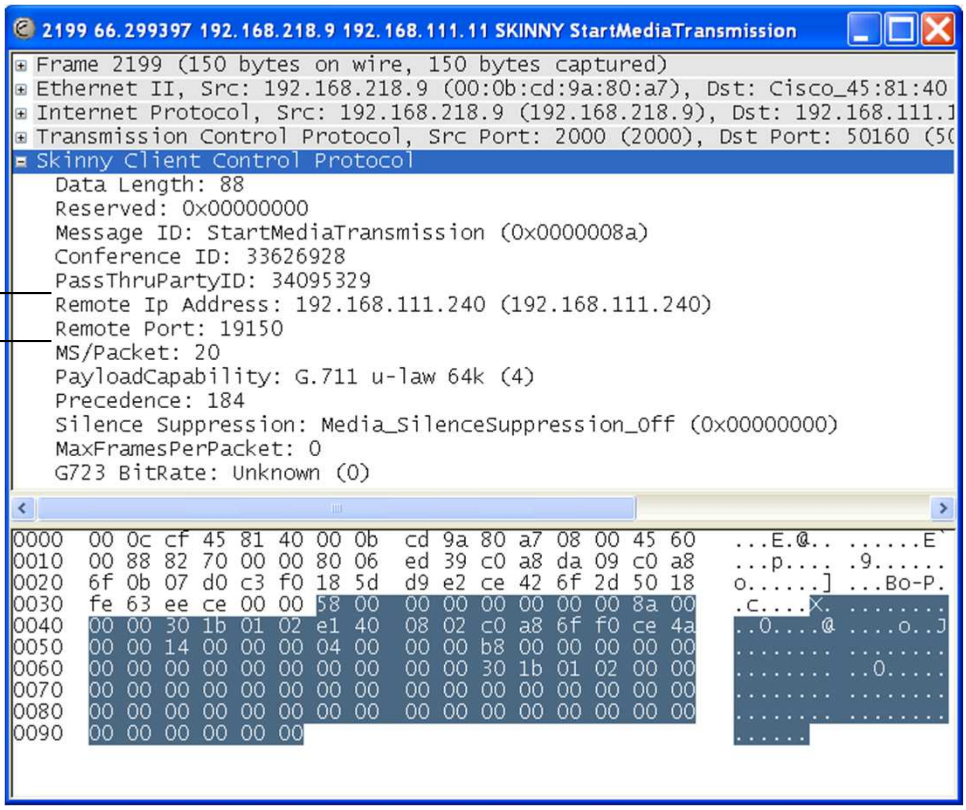
(C) 2005,2012 Technologeeks.com - All Rights Reserved




The "Start Media Transmission" is one of the more complex SCCP messages, due to its many fields. Its format is shown above, and in the following illustration.

The "Payload Capability" denotes the type of RTP transport (e.g. "4" for G.711, as we have seen for H.323). RTP is handled next.

CCM instructs phone to connect to this IP and port with RTP

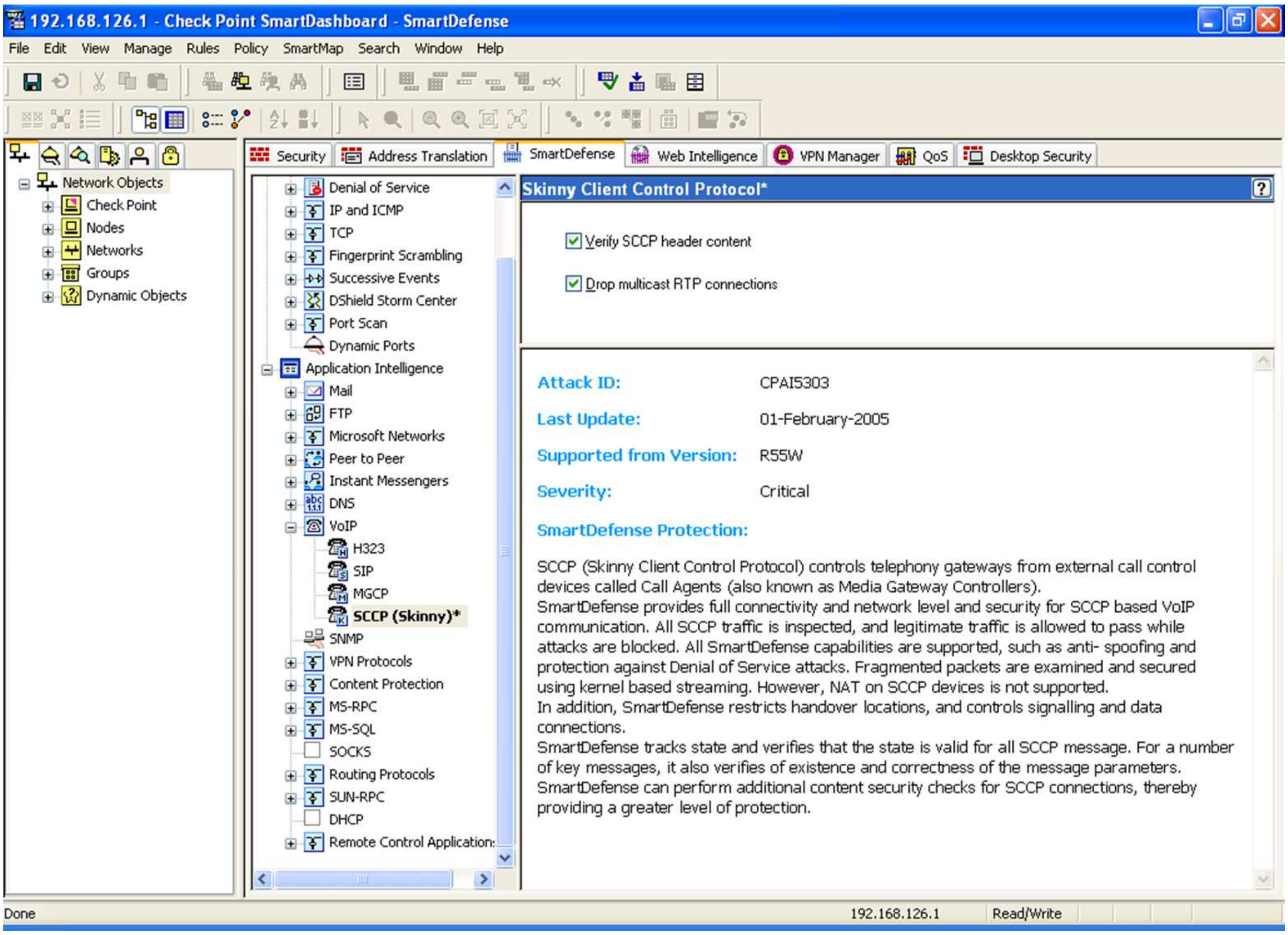



(C) 2005,2012 Technologeeks.com - All Rights Reserved

SCCP

# SCCP ("Skinny") Firewall Features

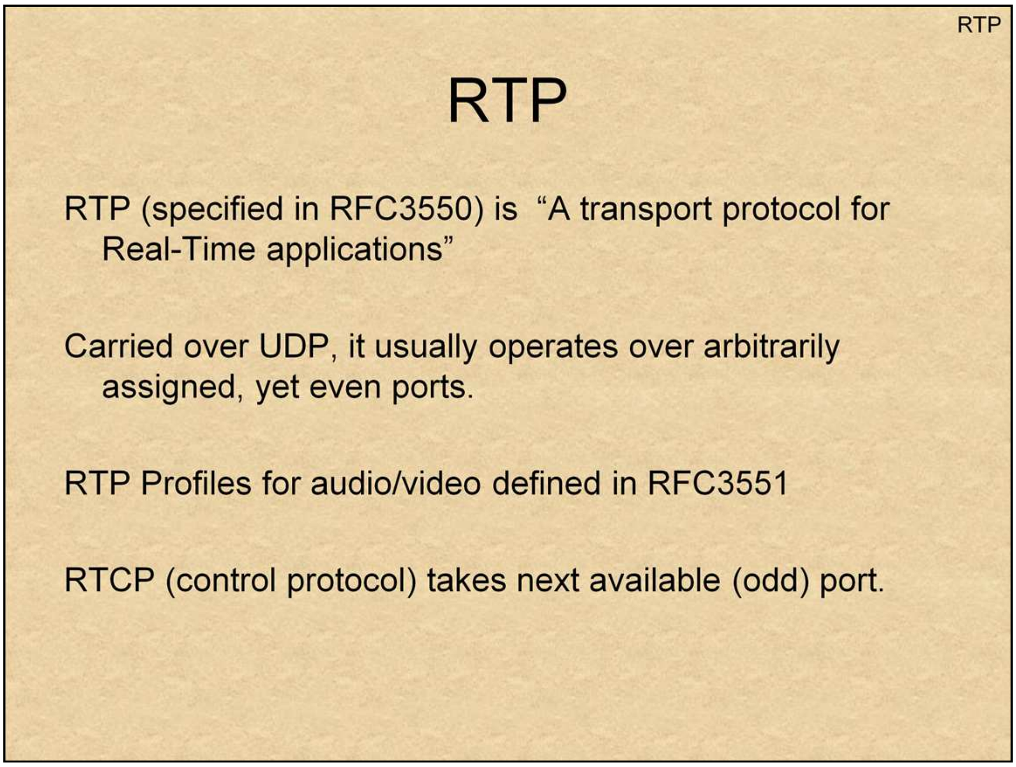
- Standard VoIP defenses
- No SCCP specific options aside from basic validation
- SCCP NAT features not supported



# RTP/RTCP

Real Time Protocol/Real Time Control Protocol

(C) 2005,2012 Technologeeks.com - All Rights Reserved



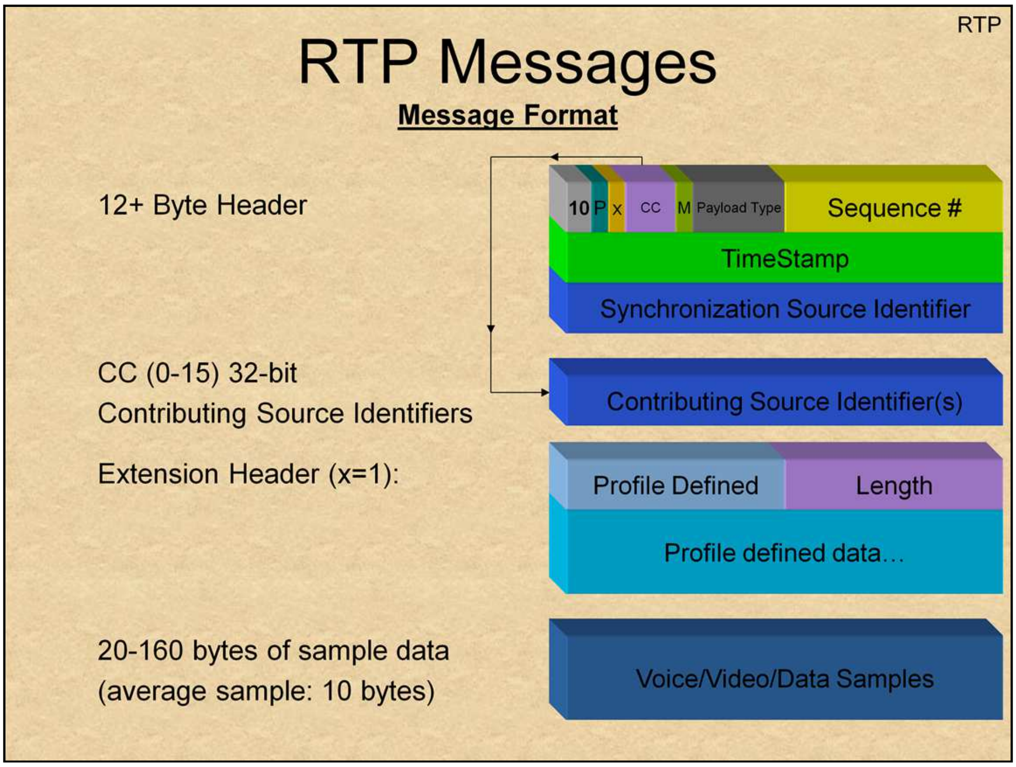

RFC1889, the original RFC for RTP, states in the abstract:

"This memorandum describes RTP, the real-time transport protocol. RTP provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services. RTP does not address resource reservation and does not guarantee quality-of- service for real-time services. The data transport is augmented by a control protocol (RTCP) to allow monitoring of the data delivery in a manner scalable to large multicast networks, and to provide minimal control and identification functionality. RTP and RTCP are designed to be independent of the underlying transport and network layers. The protocol supports the use of RTP-level translators and mixers. "

RTP is unusual in the sense that it has been adopted as part of H.323 even though it is an IETF standard. Further, it has grown to widespread acceptance as a multimedia streaming protocol outside of H.323 as well (in fact, we will see it is also handled by SIP messages).

RTP works over ephemeral ports, as set up by H.245. It further has a control component – RTCP. To distinguish between the two, the former uses even ports, and the corresponding RTCP stream – odd ones.

(C) 2005,2012 Technologeeks.com - All Rights Reserved




RTP Headers are fixed 12-byte headers, including only the necessary fields for transport management. The fields are listed below:

**Version (V- 2 bits):** fixed as a constant identifier, “2” (i.e. “10”).

**Padding (P – 1 bit):** Bit flag. When set, the packet is padded by a number of padding octets. To deal with packet, one must work backwards - The very last octet of the packet will specify the padding length. Padding is mostly used for encryption algorithms, which work with a fixed (usually 128-bit) block size.

**eXtension (X – 1 bit):** Bit flag. When set, the RTP header is followed by an extension header, as shown above. Extension headers are custom defined.

**CSRC count (CC – 4 bits):** Containing the number of Contributing Source (CSRC) identifiers following the fixed header

**Marker (M – 1 bit):** Profile defined. May be ignored (as is the case with RTCP)

**Payload type (PT – 7 bits)**: Identifying the format of the RTP payload – set by the application. Payload types are statically mapped to payload formats. As the RFC states, “RTP senders emits a single RTP payload type at any given time; this field is not intended for multiplexing separate media streams”.

**Sequence Number (16 bits):** Randomly generated sequence number, incremented by 1 for every RTP packet sent. This is intended for the same functionality as TCP’s sequence numbers – detecting packet loss, and regenerating lost packets.

**TimeStamp (32 bits):** Time stamp generated during forming of RTP packet. Clock is assumed to be a proper, monotonically increasing clock, to allow for round-trip and delay calculations. The RFC States that:

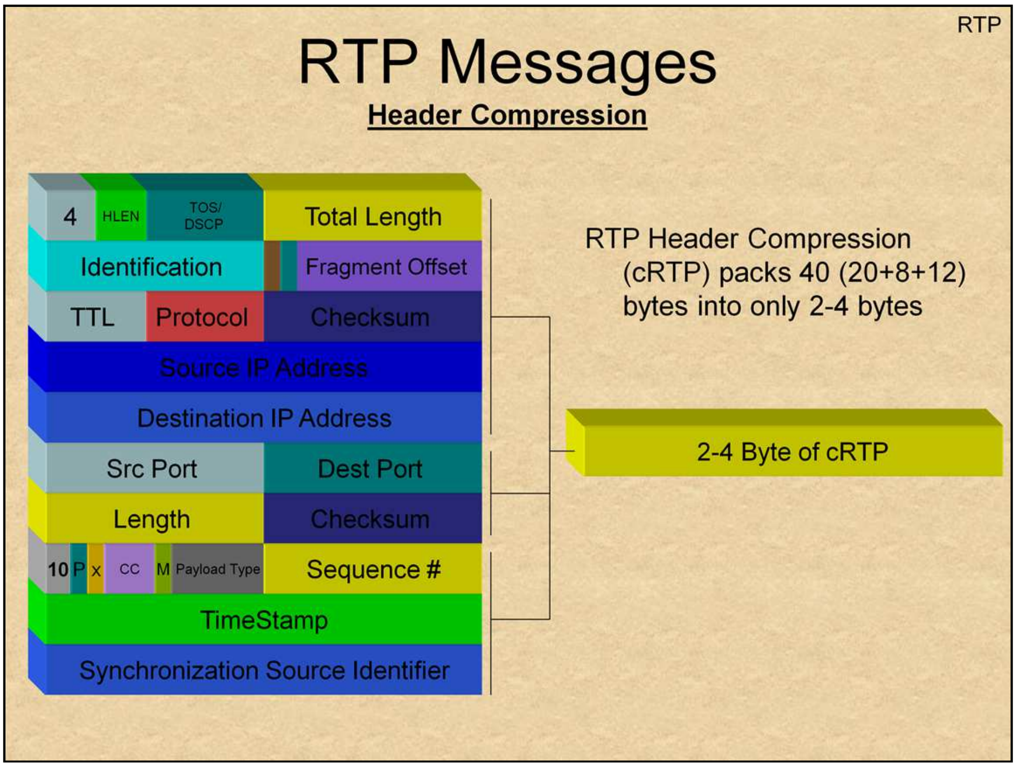
(C) 2005,2012 Technologeeks.com - All Rights Reserved

*“As an example, for fixed-rate audio the timestamp clock would likely increment by one for each sampling period. If an audio application reads blocks covering 160 sampling periods from the input device, the timestamp would be increased by 160 for each such block, regardless of whether the block is transmitted in a packet or dropped as silent. The initial value of the timestamp is random, as for the sequence number. Several consecutive RTP packets may have equal timestamps if they are (logically) generated at once, e.g., belong to the same video frame. Consecutive RTP packets may contain timestamps that are not monotonic if the data is not transmitted in the order it was sampled, as in the case of MPEG interpolated video frames. (The sequence numbers of the packets as transmitted will still be monotonic.)”*

**Synchronization Source Identifier (SSRC - 32 bits):** This field is chosen randomly, and is meant to identify the synchronization source. The chances of a random collision are slim (1 in 4,294,976,296), but RTP endpoints must be able to detect and resolve collisions.

**Contributing Source Identifier List (CSRC List - 0-15 * 32-bits):** An optional list of contributing sources. Anywhere from 0 up to 15 sources may be specified (although more than 15 sources may exist).

(C) 2005,2012 Technologeeks.com - All Rights Reserved



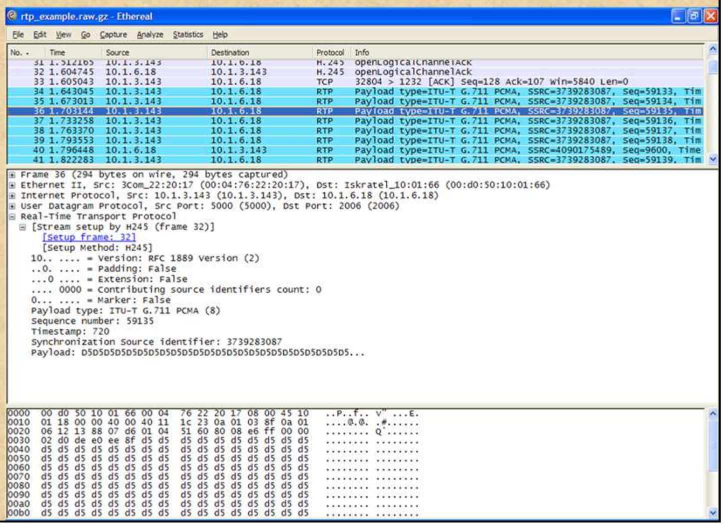

Even with a mere 12-bytes , RTP's header can be a concern. A dozen bytes may not seem like much, but along with IP (20 bytes) and UDP (8 bytes) – that's 40 bytes. In some cases, (e.g. G.729) this protocol header overhead is twice the payload. CRTP, the RTP header compression, enables to compress the header further – along with the UDP and IP headers – to a mere 2-4 byte value. This is feasible since many of the IP header fields are immutable, and certainly UDP ones are fully predictable.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

RTP

# RTP Messages

The following example illustrates an RTP message, with G.711 PCM data




To decode the G.711, any analyzer such as Ethereal can save the payload in the .au format (Sun Audio) which is readily readable by WinAmp or Windows Media Player.

To do so, simply:

1) Apply a filter to show all the RTP packets
2) Select “Analyze RTP” from Ethereal (usually under “Analysis” or “Statistics”).
3) Analyze all streams
4) Save as… (.AU file)

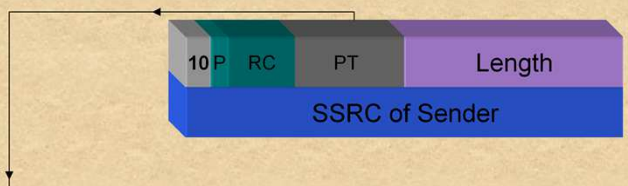
(C) 2005,2012 Technologeeks.com - All Rights Reserved

RTCP

# RTCP Messages

**Message Format**

8 Byte Header




| RTCP Message Type | PT | Purpose |
|---|---|---|
| SR (Sender Report) | 200 | transmission and reception statistics from participants that are active senders |
| RR (Receiver Report) | 201 | for reception statistics from participants that are not active senders |
| SDES (Source Description) | 202 | Source description |
| BYE | 203 | Indicates end of participation |
| APP | 204 | Application Specific Functions |

Multiple RTCP Messages may be carried in the same packet.

RTCP, The Real Time Control Protocol, works as a subset of RTP. It is meant to provide real time statistics, so as to enable QoS.

RTCP currently defines 5 messages, as shown above:

- **Sender Report (SR – PT=200):** A status message by the stream source, specifying real time connection statistics.

- **Receiver Report (RR – PT=201):** The same, coming from the receiver endpoints.

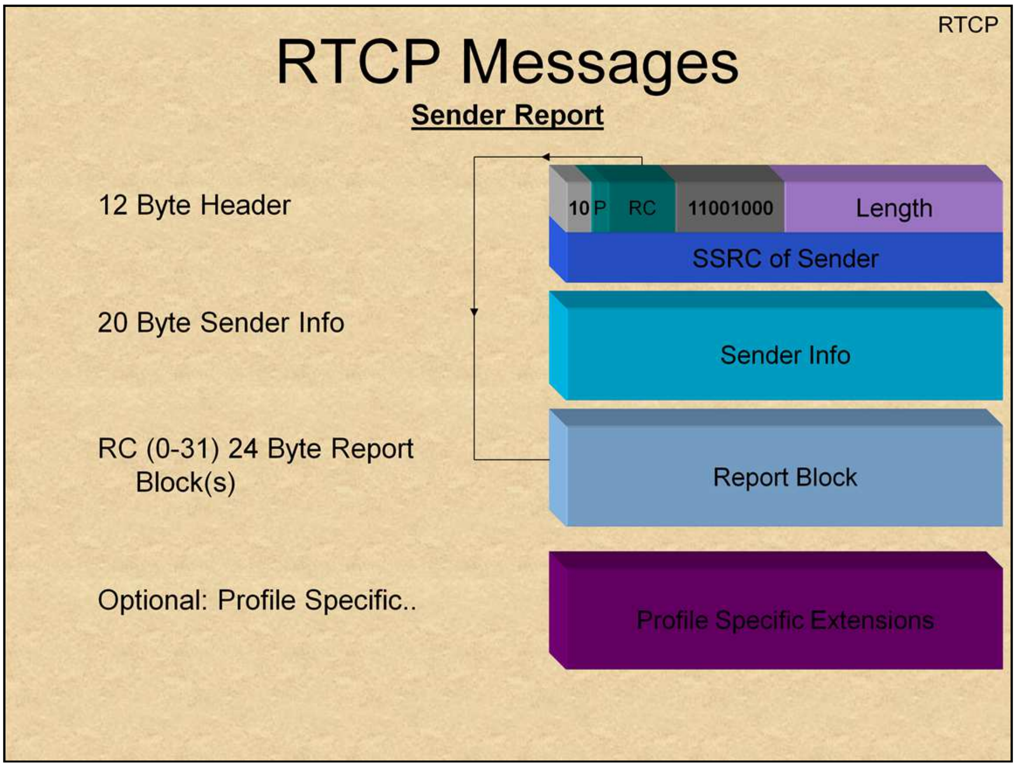
- **Source DEScription (SDES – PT=202):** Messages coming from the potential sources, describing them and identifying capabilities

- **BYE (PT=203):** A hangup message

- **APP (PT=204):** Application specific messages

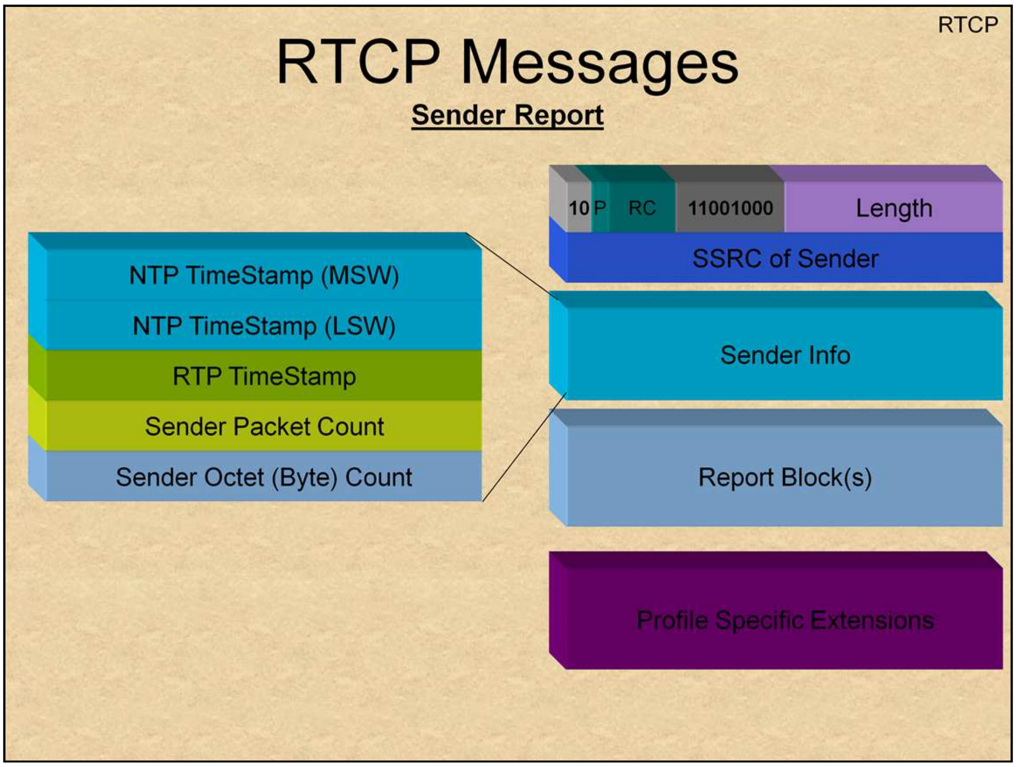
Multiple RTCP messages can be carried in one UDP packet. RTCP messages may be encapsulated in the packet, one after the header, with a new RTCP header for each message.

(C) 2005,2012 Technologeeks.com - All Rights Reserved




RTCP's "Sender Report" is comprised of a fixed 20-byte "Sender Info" block, and an optional 0-31 (5-bit value – RC) "Report Blocks". Each report block is 24 bytes.

(C) 2005,2012 Technologeeks.com - All Rights Reserved




The 20 bytes of the "Sender Info" are mandatory, and contain the following:

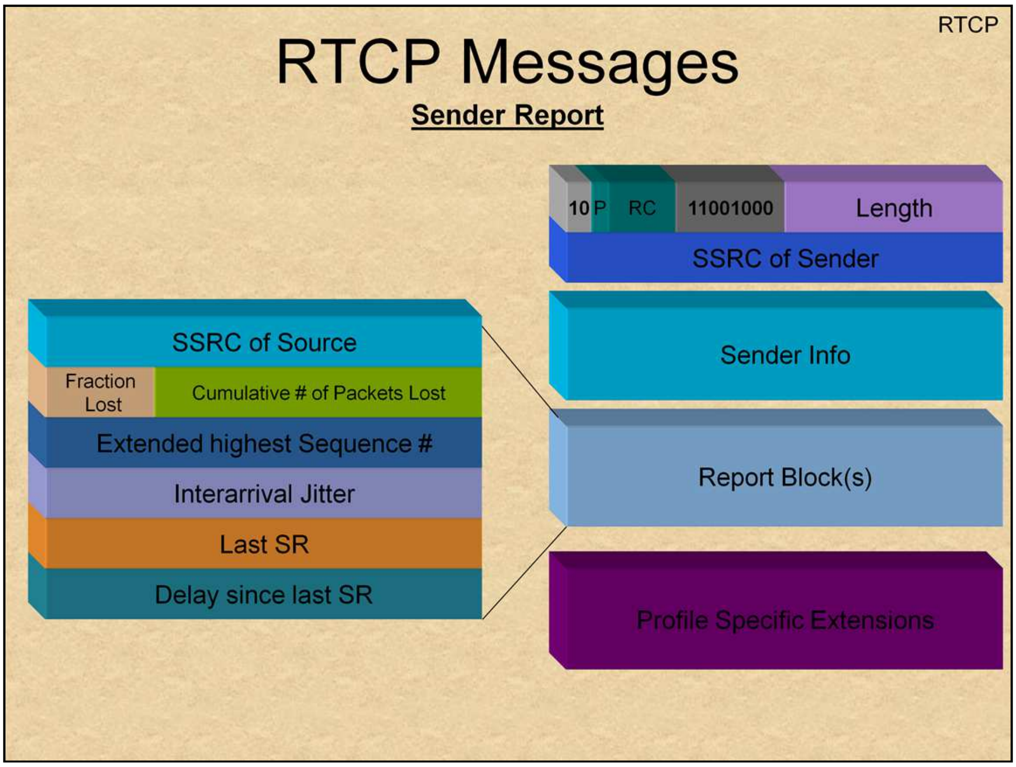
- **NTP TimeStamp:** An accurate (64-bit) timestamp, used for synchronization purposes. The timestamp is assumed to be from an accurate paced clock, though not necessarily sync'ed with other clocks.

- **RTP TimeStamp:** a 32-bit timestamp that is started from when the RTP stream first started.

- **Sender Packet Count:** a 32-bit count of RTP packets

- **Sender Octet Count:** a 32-bit count of octets (bytes) transmitted in the above packets.

(C) 2005,2012 Technologeeks.com - All Rights Reserved




The “Report Blocks” provide more details as to the source, and allow the determination of network lag and round-trip time. Each is 24 bits, as follows:

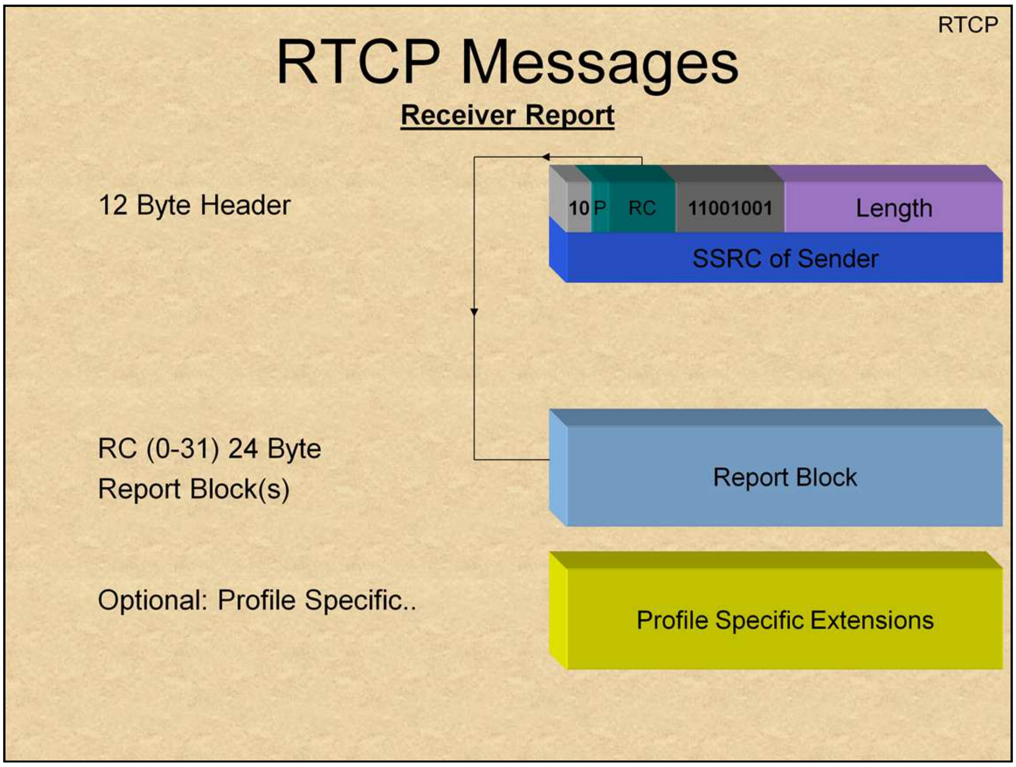
- **SSRC of Source:** Synchronization Source Identifier of source reporting this. Note that this may differ than the “SSRC of Sender”, above.
- **Fraction Lost:** a floating point value denoting the percentage of packets lost.
- **Cumulative # of Packets Lost:** the actual number of packets lost – 24 bits.
- **Extended highest sequence #:** Highest sequence # received (16-bits) + count of Seq# cycles
- **Interarrival Jitter:** measured in timestamp units – mean deviation of packet reception vs. packet sending

$$D_{(i,j)} = (R_j - R_i) - (S_j - S_i) = (R_j - S_j) - (R_i - S_i)$$
$$J_{(i)} = J_{(i-1)} + (|D_{(i-1,i)}| - J_{(i-1)})/16$$

Where $R_i$ = Receive time of packet i
$S_i$ = Timestamp of packet i

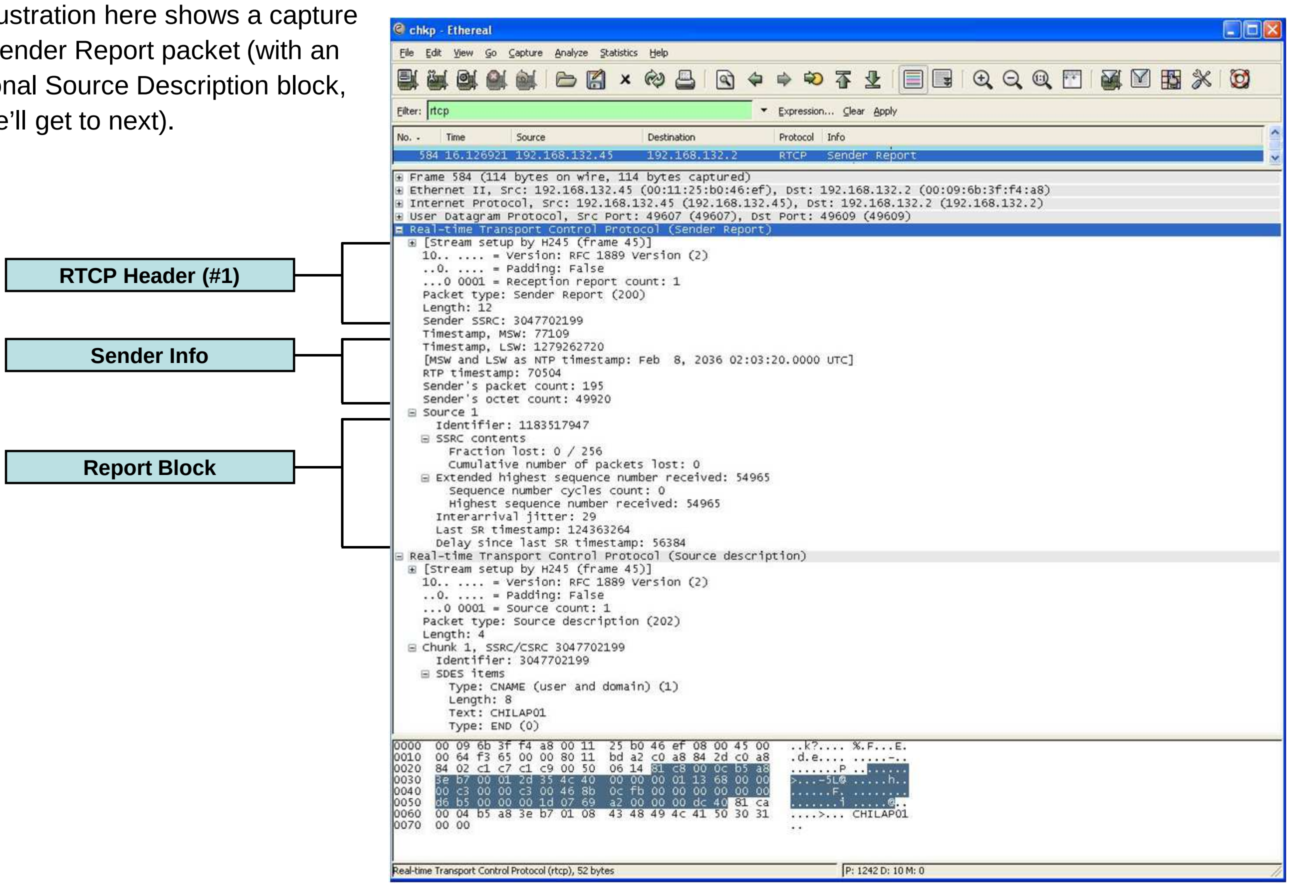
- **Last Sender Report:** Timestamp – middle 32 bits of the 64 bits of the NTP in the last SR.
- **Delay since Last Sender Report:** in units of 1/65536 seconds, between reception of last SR, and this report block.


(C) 2005,2012 Technologeeks.com - All Rights Reserved




The Receiver Report (PT = 201) is identical to the Sender Report packet, with the exception that the Sender Info Block is omitted.

The illustration here shows a capture Of a Sender Report packet (with an additional Source Description block, that we'll get to next).



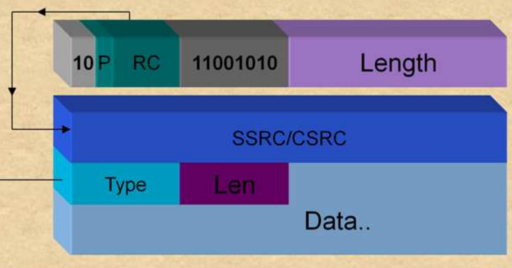
RTCP

# RTCP Messages

## Source Description

4 Byte Header

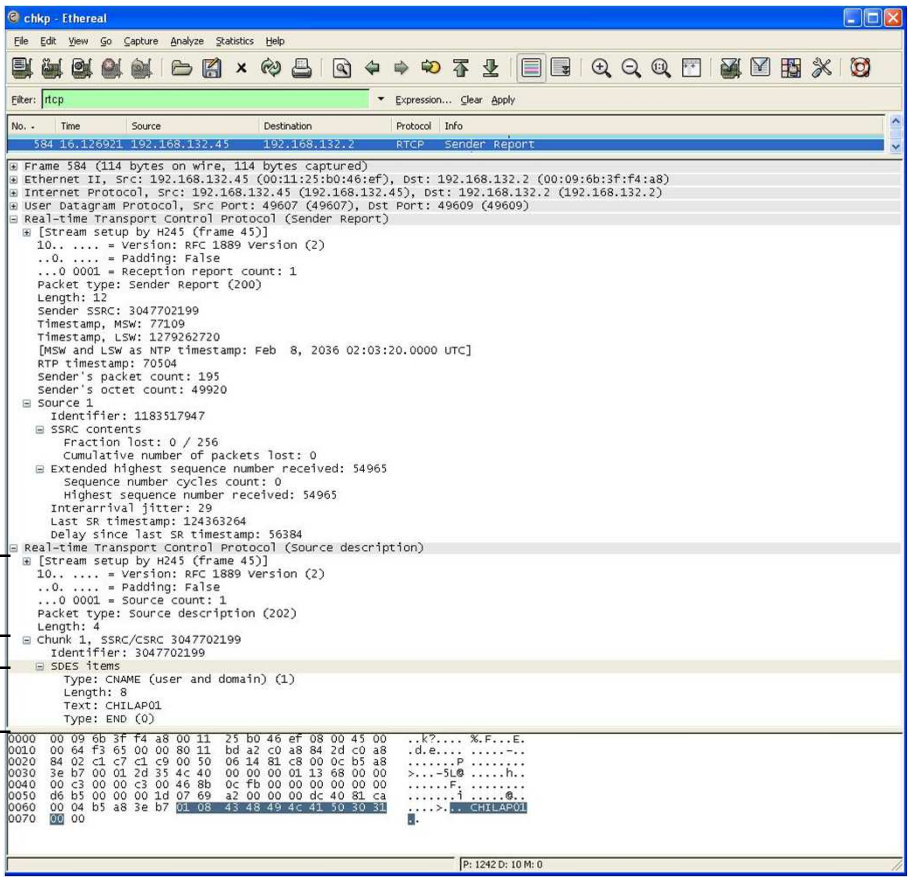
SC (0-31) variable length SDES Chunks




| SDES Data Item | Type | Purpose |
|---|---|---|
| CNAME | 1 | Canonical Name of Source |
| NAME | 2 | Common (display) Name of Source |
| EMAIL | 3 | Email Address of Source |
| PHONE | 4 | Phone # of Source in international format (+ 1 800 ..) |
| LOC | 5 | Geographical Location of Source |
| TOOL | 6 | Application/Tool used by Source |
| NOTE | 7 | Notice message, Status, etc (Textual) |
| PRIV | 8 | Private Extension – Application Specific |

The Source DEScription (SDES) packets provide miscellaneous ancillary data about the source of the RTP stream. This is mostly for human purposes – providing the source's ID, phone #, location, etc. The table above summarizes the defined record types.

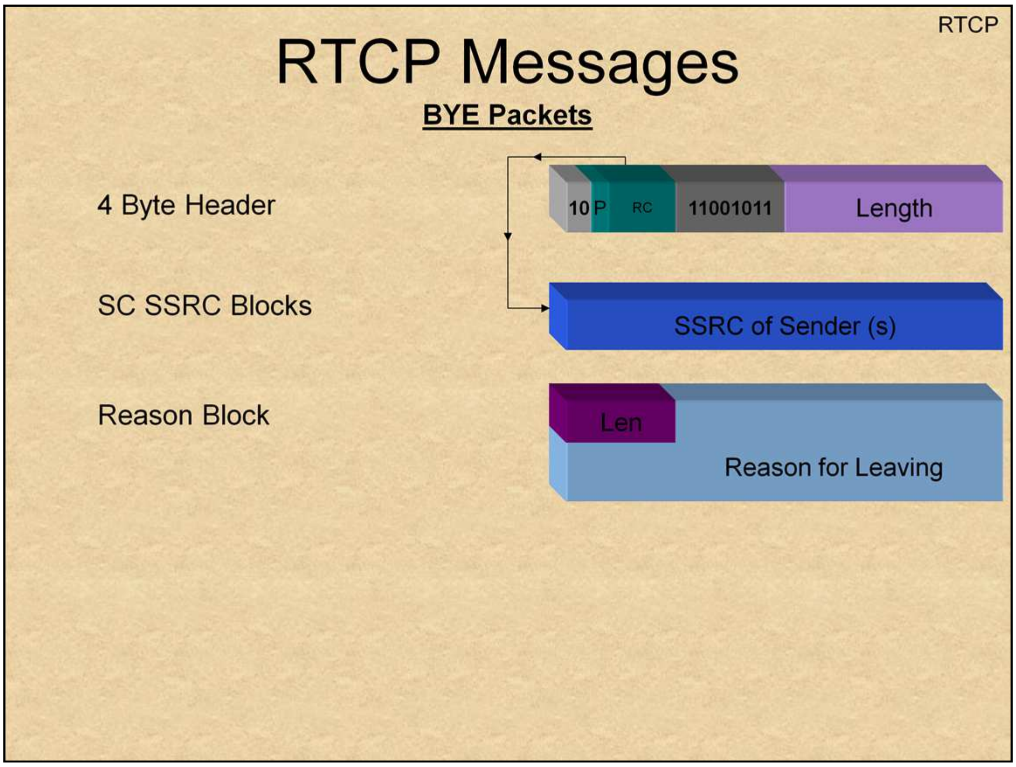
The illustration on this page is the same packet capture as the one on the last page, but this time we focus on the second RTCP Header, with the SDES Block:




RTCP Header (#2)

SDES Block, type 1

(C) 2005,2012 Technologeeks.com - All Rights Reserved




(C) 2005,2012 Technologeeks.com - All Rights Reserved

RUDP

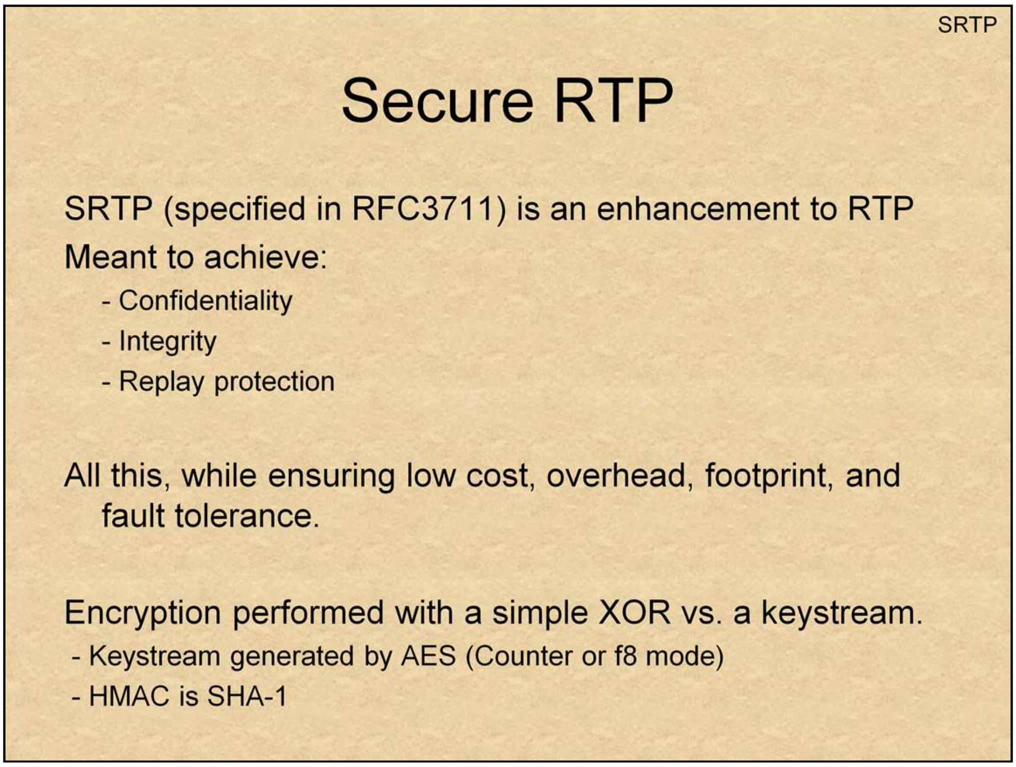
# RUDP

- RFC908 (v1) and RFC1151(v2)
- Similar to "plain" UDP, but more reliable
- Employs a principle of Forward Error Correction:
  Packets are sent in multiple copies
- Used extensively by CISCO, not an industry standard

RUDP is an effort to make UDP, the User Datagram Protocol, a reliable one, at the cost of bandwidth.

Cisco is known to use RUDP (implementing Q.931). A somewhat similar approach is assumed to be used by Skype.

(C) 2005,2012 Technologeeks.com - All Rights Reserved



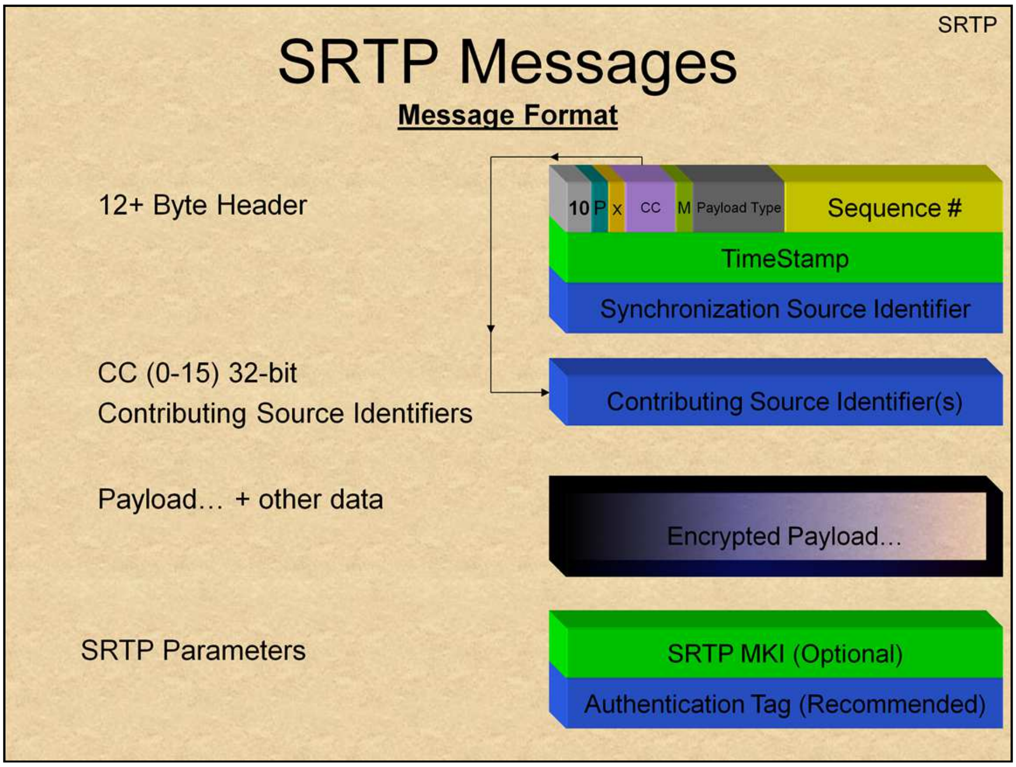

In order to provide security for RTP and RTCP, a proposed extension, known as **Secure RTP (SRTP)** specified in **RFC3711**, has been proposed.

This solution is meant to provide strong encryption and authentication to these protocols, without hampering their critical real time performance.

To provide for speed, encryption is a simple XOR operation, with a precomputed keystream. The encryption algorithm of choice is AES, and digital signatures (HMAC) are applied by using the NIST approvded SHA-1 hash function.

Performance wise, this is far more efficient than IPSec, as IPSec encrypts on a per packet basis.

(C) 2005,2012 Technologeeks.com - All Rights Reserved




SRTP encrypts the RTP payload, and authenticates the entire packet. It piggybacks over normal RTP by appending two fields to an RTP message:

**SRTP MKI:** Identifies master key from which session encryption key was derived
**Authentication Tag:** Message Authentication code

A cryptographic context is uniquely identified by the triplet context identifier:
context id = <SSRC, destination network address, destination transport port number>

(C) 2005,2012 Technologeeks.com - All Rights Reserved

# SIP

## Session Initiation Protocol

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SIP

# SIP

The **Session Initiation Protocol** (v2 - RFC3261) is a highly extensible generic session management protocol.

SIP Establishes presence, and enables mobility.

Does not provide services, but primitives for services.

Modeled after HTTP, with a different set of methods.

Transport: UDP (v1.0), SCTP, TCP(v2.0). TLS Optional

The **Session Initiation Protocol**, commonly known as “**SIP**” is a relatively new protocol, that is fast emerging as the new standard for VoIP, and multimedia sessions in general. It is a scalable, agile protocol, that is session agnostic, and can handle any type of media.

SIP Was originally drafted around 1996. It matured to version 1.0 with **RFC2543**, but was later revised to version 2.0 (**RFC3261**), which is the suggested standard. The protocol was greatly enhanced in between the versions.

Originally, SIP worked primarily with UDP. Version 2.0, however, works over either UDP or TCP, preferably the latter. UDP usage is somewhat limited as there is no support for fragmentation, and SIP messages must not exceed the PMTU. There is a compact form for SIP messages (more on that later), but that generally doesn’t help much when messages with longer bodies are involved.

RFC3261 also added support for Transport Layer Security (TLS – RFC 2246), allocating port 5061 for SIP over TLS. An extension to SIP allows transport over the Stream Control Transmission Protocol (SCTP – RFC2960).

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SIP

# SIP

SIP supports five facets of session management:

**User Location**: Where is user currently located?

**User Availability**: Is user free to start/join a session?

**User Capability**: Endpoint capability (voice/video/etc..)

**Session Setup**: “Ringing” and “picking up”

**Session Management**: Modifying/Terminating Sessions

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SIP

# Advantages of SIP

SIP boasts several important advantages over H.323

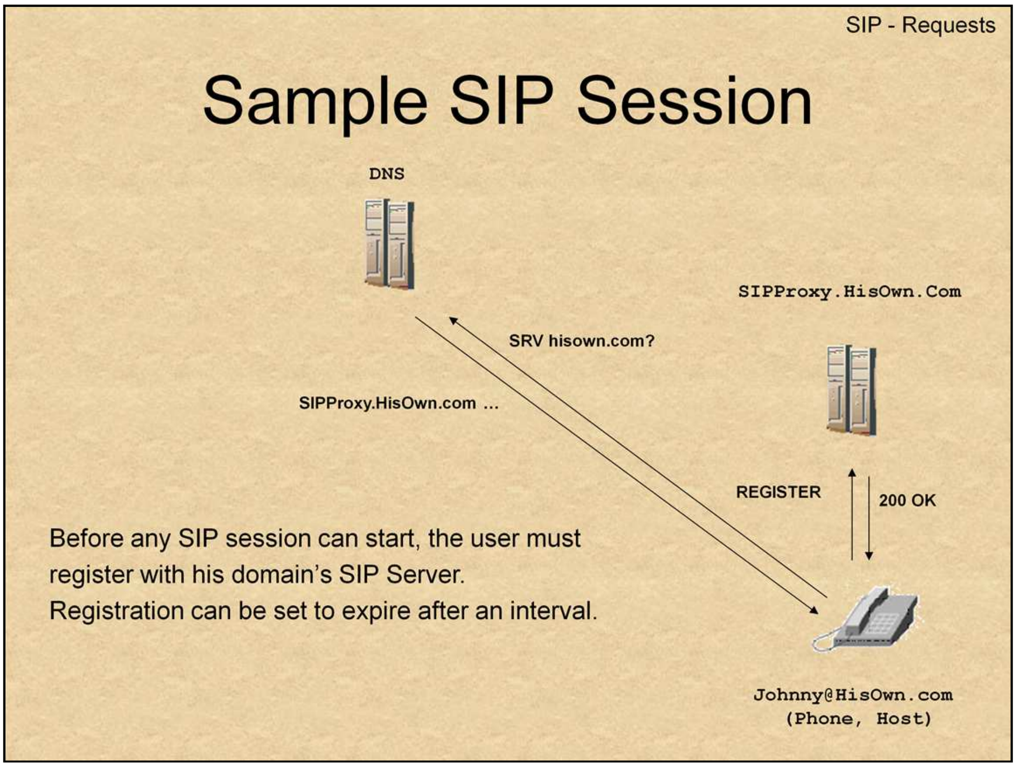
**Open Protocol**: Well documented, in RFCs

**Plain Text messages**: as opposed to H.323's ASN.1

**Easily extensible**: by means of additional messages, etc.

**Scalable:** as effective in small networks as in large ones

(C) 2005,2012 Technologeeks.com - All Rights Reserved




**Example**:

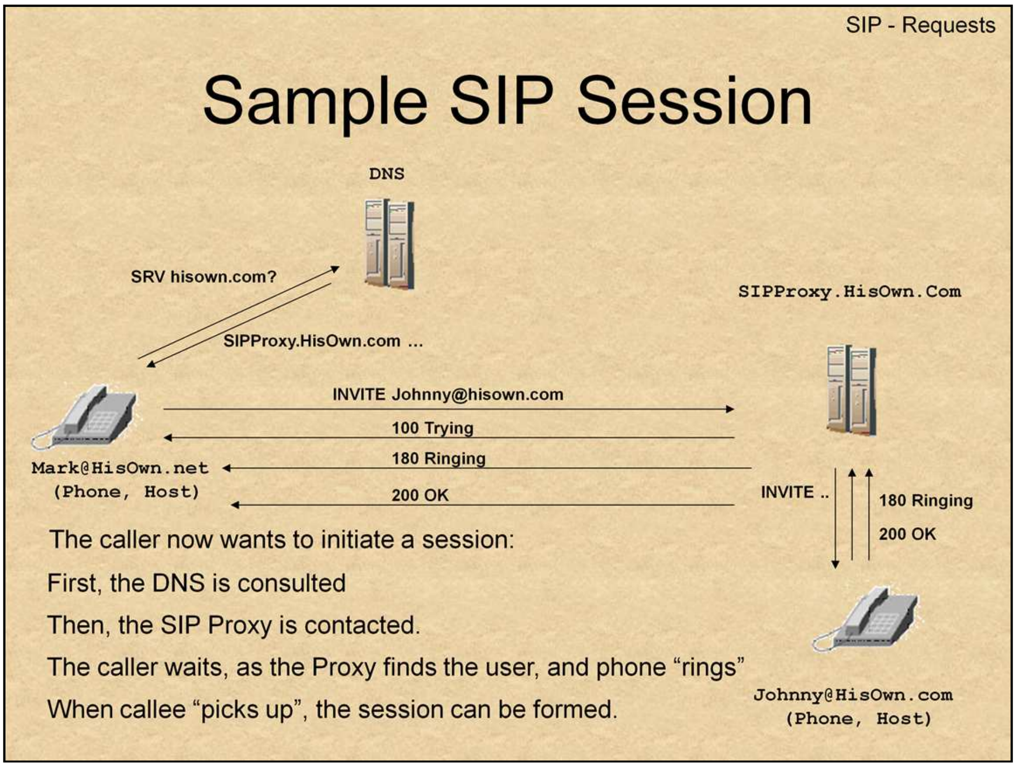
```
REGISTER sip:JsIPPhone.HisOwn.com SIP/2.0
Via: SIP/2.0/UDP SIPProxy.Hisown.com:5060
Max-Forwards: 10
To: JL <sip:JL@HisOwn.com>
From: JL <sip:JL@HisOwn.com>
Call-ID: 3242534554645656536
CSeq: 1234 REGISTER
Contact: <sip:JL@212.150.77.17>
Expires: 36000
Content-Length: 0
```

The registration expires after 10 hours (i.e. 36000 seconds)

An "OK" response confirms registration:

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP SIPProxy.HisOwn.com:5060; received=212.150.77.17
To: JL <sip:JL@HisOwn.com>
From: JL <sip:JL@HisOwn.com>
Call-ID: 3242534554645656536
CSeq: 1234 REGISTER
Contact: <sip:JL@192.0.2.4>
Expires: 36000
Content-Length: 0
```

(C) 2005,2012 Technologeeks.com - All Rights Reserved




Now, at some point, someone will attempt to contact our hero. This is done in a manner not unlike EMail transfer:

First, a DNS is consulted. Again, an SRV record is looked up. This will reveal the address of the SIP Proxy, with which the callee has previously registered.

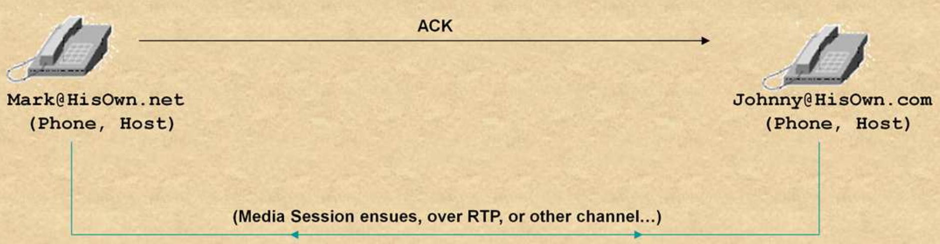
It's important to emphasize there may be more than one proxy in between the peers; For example, a common setup is corporate networks is to have an outgoing SIP Proxy for calls. The RFC calls this a "Trapezoid" setup (Caller->Proxy->Proxy->Callee). Just like HTTP requests can be forwarded via proxies, the same would work here.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SIP - Requests

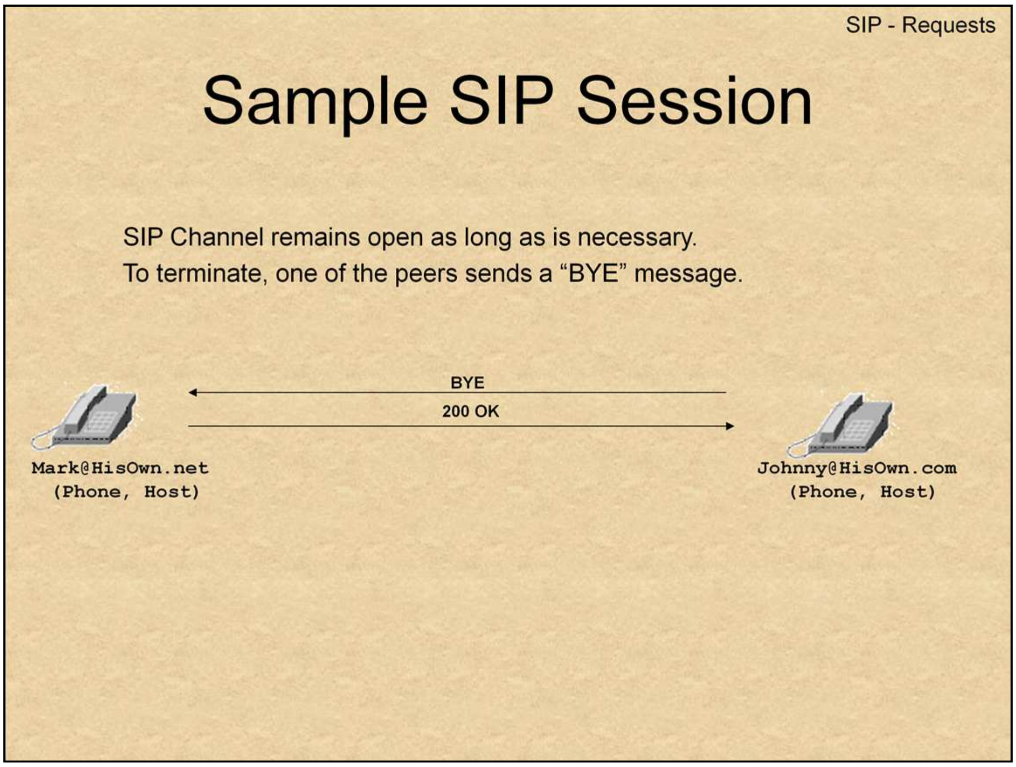
# Sample SIP Session

After callee "picks up", a peer to peer control channel is formed
By sending an "ACK" message directly from the caller to the callee.



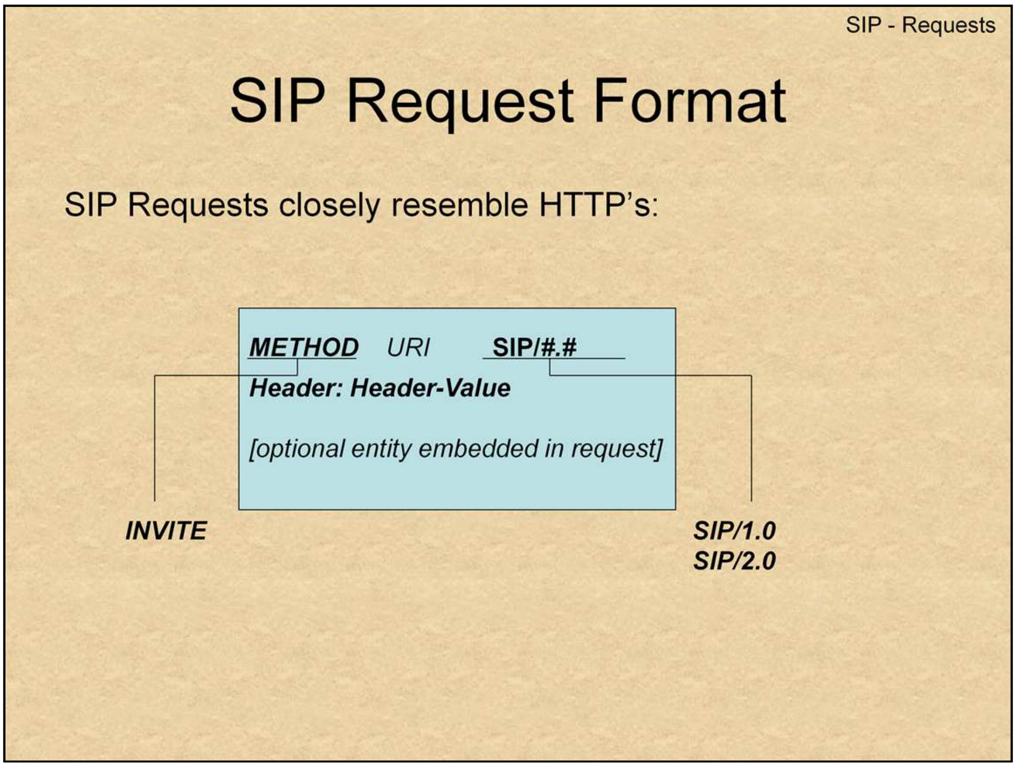

The peers establish a session as described by the SIP payload.


(C) 2005,2012 Technologeeks.com - All Rights Reserved




The “BYE” command tears down the session. (Quite simple)

(C) 2005,2012 Technologeeks.com - All Rights Reserved




SIP syntax is essentially the same as HTTP's:

The **Method** is a verb denoting the request type (e.g. a "GET" request, a "PUT", etc). The next slide elaborates on the methods used in HTTP.

The **U**niform **R**esource **I**dentifier, or **URI**, specifies the unique identifier of the file or object in question. This is commonly a URL (Uniform Resource Locator).

The SIP Version (**SIP/1.0** or **SIP/2.0**) follow the URI. A request can be simple enough to end here, or be followed by additional header fields.

Header fields are in the form:

**Header: value**

Each header field is specified on one line, terminated by a CRLF. The request itself is terminated by a double CRLF sequence.

The **Entity** is defined as the payload of the request, or reply. It is strictly optional ; Normal GET and HEAD requests, as we will see, have no entities associated with them. POST and PUT, however, do.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SIP - Requests

# SIP Request Methods

RFC3261 defines the core 6 methods:

| Method Name | Usage |
|---|---|
| REGISTER | Register presence with SIP Server |
| INVITE | Invite remote user to start a session |
| ACK | Initiating peer to peer connection |
| CANCEL | Cancel Invitation |
| OPTIONS | Query available options |
| BYE | Terminate Session |

As we have seen, Each SIP Request contains one of the methods described above.

**REGISTER** requests the primary registration of the User Agent with the SIP Proxy server.

**INVITE** requests attempt to initiate a session between two peers, with or without a proxy.

**ACK** is sent as a reply to confirm that an invitation has been accepted.

The **CANCEL** method is used to cancel an invitation in progress.

**OPTIONS** is used, as in HTTP, to query Server Options

Finally, when a SIP session is to be terminated, the **BYE** method is used.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SIP - Requests

# SIP Request Methods

7 Further SIP Methods are defined in other RFCs:

| Method Name | Defined in | Usage |
|---|---|---|
| INFO | RFC2976 | Application Level information |
| MESSAGE | RFC3428 | Extension for Instant Message |
| NOTIFY | RFC3265 | Event notification |
| PRACK | RFC3262 | Provisional (1xx) Message Ack |
| SUBSCRIBE | RFC3265 | Subscribe to event notifications |
| REFER | RFC3515 | Refer to external resources |
| UPDATE | RFC3311 | Updating Session Parameters |

Other RFCS, as the ones shown above, enhance SIP with seven other methods. These requests are not part of the original SIP standard, and as such may or may not be supported by an application.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SIP - Requests

# SIP Headers

Methods may be modified by specific headers:

| Header Name | Usage |
|---|---|
| Accept[-encoding]<br>[-Language] | Limit Content Encodings (for reason phrases)<br>Limit Language used (for reason phrases) |
| Alert-Info | Specify Ring-Tone for UAS(!) |
| Allow | Specify allowed methods for UA |
| Authentication-Info | Digest Authentication, as per RFC2617 |
| Authorization | As per HTTP/1.1 |
| Call-ID | Specify unique ID for call (e.g. Conferences) |
| Call-Info | For Display purposes (JPGs, textual description) |
| Contact | Entity in charge of this call. (Contact Info) |

SIP Methods are modified by the above headers, as shown in this, and the next slides.
The gray header fields are the one that are mandatory in all SIP requests, as stated by these are:

| Header Field | Purpose |
|---|---|
| Call-ID | Globally unique identifier for call |
| CSeq | Unique identifier for transaction (32-bits) |
| From | Logical identity of initiator |
| Max-Forwards | Max # Hops before 483 rejection (recommended: 70) |
| To | Logical identity of recipient |
| Via | Transport protocols chosen |

Additionally, for SIP connection requests, the Contact: Header must be present, as well.
The table on pages 66-67 shows the relation between the Header fields and the requests.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SIP - Requests

# SIP Headers

| Header Name | Usage |
|---|---|
| Content-[Disposition] | Specify "Alert", "Icon","Session" or "Render" |
| [Encoding] | Specify Content Encodings (e.g. gzip) |
| [Language] | Specify Language used (same as HTTP – RFC2616) |
| [Length] | Specify Content Length (0 = Empty message) |
| [Type] | Specify Content Type (See Below) |
| CSeq | Uniquely identifies (and orders) transactions |
| Date | Date/Time specification |
| Error-Info | Additional Info on errors |
| Expires | Method dependent |
| From | Originator of this request |
| In-Reply-To | If this response is a reply, specifies Call-ID of request |
| Max-Forwards | Maximum # of SIP gateways/proxies allowed |
| Min-Expires | Minimum Refresh Time |

Possible Content-Types are:

**text/plain:** Plain text
**text/html:** HTML text
**application/sdp:**: SDP (for ACK, INVITE, or UPDATE)
**message/sipfrag:** SIP fragments (for NOTIFY)
**application/xml+dialog:** XML dialog
**application/xml+conf:** XML conference info
**application/cpim:** Common Presence & Instant Messaging
**application/isup:** Encapsulated ISUP in INVITE, BYE, or INFO (RFC3204)

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SIP - Requests

# SIP Headers

| Header Name | Usage |
|---|---|
| MIME-Version | "1.0", for MIME encoded requests |
| Organization | optional Organization textual description |
| Priority | "non-urgent"… "emergency" |
| Proxy-Authenticate | Proxy Authentication blob, if required |
| Proxy-Require | Directive to proxies, for required SIP extensions |
| Record-Route | Forces Proxies to record their presence |
| Reply-To | Specify alternate address to "From", for possible replies |
| Require | Directive to UA, for required SIP extensions |
| Retry-After | If call cannot be completed (e.g. Busy), set retry period |
| Route | Require a specific proxy |
| Server | Specify server version |

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SIP - Requests

# SIP Headers

| Header Name | Usage |
|---|---|
| Subject | Specify subject of call |
| Supported | Supported SIP Extensions of the UA |
| TimeStamp | Specifies when UA sent request |
| To | Logical Recipient (Display name or other SIP URL) |
| Unsupported | UNSupported SIP Extensions for the UA |
| User-Agent | UA Identification |
| Via | Specifies transport protocols used |
| Warning | See Below |
| WWW-Authenticate | Prompts for authorization |

SIP **Warnings** are 3xx error codes:

Warnings 300 through 329 are reserved for indicating problems with keywords in the session description,

330 through 339 are warnings related to basic network services requested in the session description

370 through 379 are warnings related to quantitative QoS parameters requested in the session description,

390 through 399 are miscellaneous warnings that do not fall into one of the above categories.

**300 Incompatible network protocol:** One or more network protocols contained in the session description are not available.

**301 Incompatible network address formats:** One or more network address formats contained in the session description are not available.

**302 Incompatible transport protocol:** One or more transport protocols described in the session description are not available.

**303 Incompatible bandwidth units:** One or more bandwidth measurement units contained in the session description were not understood.

**304 Media type not available:** One or more media types contained in the session description are not available.

**305 Incompatible media format:** One or more media formats contained in the session description are not available.

**306 Attribute not understood:** One or more of the media attributes in the session description are not supported.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

**307 Session description parameter not understood**: A parameter other than those listed above was not understood.

**330 Multicast not available:** The site where the user is located does not support multicast.

**331 Unicast not available:** The site where the user is located does not support unicast communication (usually due to the presence of a firewall).

**370 Insufficient bandwidth:** The bandwidth specified in the session description or defined by the media exceeds that known to be available.

**399 Miscellaneous warning:** The warning text can include arbitrary information to be presented to a human user or logged. A system receiving this warning MUST NOT take any automated action.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SIP - Requests

# SIP Header Compact Forms

| Header Name | Compact Form |
|---|---|
| Call-ID | i |
| Contact | m |
| Content-Encoding | e |
| Content-Length | l |
| Content-Type | c |
| Event | o (NOTIFY – RFC3265) |
| Refer-To | r (REFER – RFC3515) |
| Referred-By | b (REFER – RFC3515) |
| Reject-Contact | j (Internet Draft) |
| Subject | s |
| To | t |
| Via | V |

Common SIP Headers may be abbreviated with single letter, as shown in the above table. This is optional.

The following tables, taken from the SIP RFC, show the usage of headers in SIP requests. The following legend is defined for the “where” field:

**R**: header field may only appear in requests;

**r**: header field may only appear in responses;

**2xx, 4xx, etc.:** A numerical value or range indicates response codes with which the header field can be used;

**c:** header field is copied from the request to the response. An empty entry in the "where" column indicates that the header field may be present in all requests and responses.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

**For each method:**

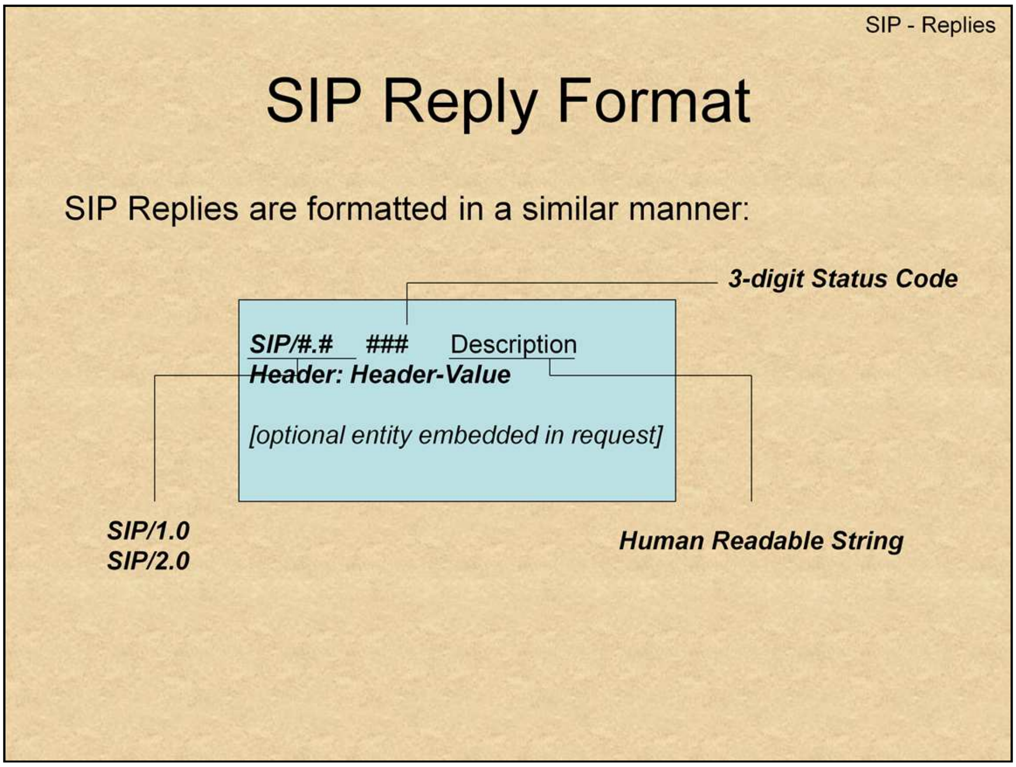
| | |
|---|---|
| **c**: | Conditional; requirements on the header field depend on the context of the message. |
| **m**: | Mandatory |
| **m***: | The header field SHOULD be sent, but clients/servers need to be prepared to receive messages without that header field |
| **o**: | Optional |
| **t**: | The header field SHOULD be sent, but clients/servers need to be prepared to receive messages without that header field. If a stream-based protocol (such as TCP) is used as a transport, then the header field MUST be sent. |
| *****: | Required only if message body is not empty |
| **-**: | Header field not applicable with this request |

| Header field | where | proxy | ACK | BYE | CAN | INV | OPT | REG |
|---|---|---|---|---|---|---|---|---|
| Accept | R | | - | o | - | o | m* | o |
| Accept | 2xx | | - | - | - | o | m* | o |
| Accept | 415 | | - | c | - | c | c | c |
| Accept-Encoding | R | | - | o | - | o | o | o |
| Accept-Encoding | 2xx | | - | - | - | o | m* | o |
| Accept-Encoding | 415 | | - | c | - | c | c | c |
| Accept-Language | R | | - | o | - | o | o | o |
| Accept-Language | 2xx | | - | - | - | o | m* | o |
| Accept-Language | 415 | | - | c | - | c | c | c |
| Alert-Info | R | ar | - | - | - | o | - | - |
| Alert-Info | 180 | ar | - | - | - | o | - | - |
| Allow | R | | - | o | - | o | o | o |
| Allow | 2xx | | - | o | - | m* | m* | o |
| Allow | r | | - | o | - | o | o | o |
| Allow | 405 | | - | m | - | m | m | m |
| Authentication-Info | 2xx | | - | o | - | o | o | o |
| Authorization | R | | o | o | o | o | o | o |
| Call-ID | c | r | m | m | m | m | m | m |
| Call-Info | | ar | - | - | - | o | o | o |
| Contact | R | | o | - | - | m | o | o |
| Contact | 1xx | | - | - | - | o | - | - |
| Contact | 2xx | | - | - | - | m | o | o |
| Contact | 3xx | d | - | o | - | o | o | o |
| Contact | 485 | | - | o | - | o | o | o |
| Content-Disposition | | | o | o | - | o | o | o |
| Content-Encoding | | | o | o | - | o | o | o |
| Content-Language | | | o | o | - | o | o | o |
| Content-Length | | ar | t | t | t | t | t | t |
| Content-Type | | | * | * | - | * | * | * |
| CSeq | c | r | m | m | m | m | m | m |
| Date | | a | o | o | o | o | o | o |
| Error-Info | 300-699 | a | - | o | o | o | o | o |
| Expires | | | - | - | - | o | - | o |
| From | c | r | m | m | m | m | m | m |
| In-Reply-To | R | | - | - | - | o | - | - |
| Max-Forwards | R | amr | m | m | m | m | m | m |
| Min-Expires | 423 | | - | - | - | - | - | m |
| MIME-Version | | | o | o | - | o | o | o |
| Organization | | ar | - | - | - | o | o | o |

(C) 2005,2012 Technologeeks.com - All Rights Reserved

| Header field | where | proxy | ACK | BYE | CAN | INV | OPT | REG |
|---|---|---|---|---|---|---|---|---|
| Priority | R | ar | - | - | - | o | - | - |
| Proxy-Authenticate | 407 | ar | - | m | - | m | m | m |
| Proxy-Authenticate | 401 | ar | - | o | o | o | o | o |
| Proxy-Authorization | R | dr | o | o | - | o | o | o |
| Proxy-Require | R | ar | - | o | - | o | o | o |
| Record-Route | R | ar | o | o | o | o | o | - |
| Record-Route | 2xx,18x | mr | - | o | o | o | o | - |
| Reply-To | | | - | - | - | o | - | - |
| Require | | ar | - | c | - | c | c | c |
| Retry-After | 404,413,480,486 | | - | o | o | o | o | o |
| | 500,503 | | - | o | o | o | o | o |
| | 600,603 | | - | o | o | o | o | o |
| Route | R | adr | c | c | c | c | c | c |
| Server | r | | - | o | o | o | o | o |
| Subject | R | | - | - | - | o | - | - |
| Supported | R | | - | o | o | m* | o | o |
| Supported | 2xx | | - | o | o | m* | m* | o |
| Timestamp | | | o | o | o | o | o | o |
| To | c(1) | r | m | m | m | m | m | m |
| Unsupported | 420 | | - | m | - | m | m | m |
| User-Agent | | | o | o | o | o | o | o |
| Via | R | amr | m | m | m | m | m | m |
| Via | rc | dr | m | m | m | m | m | m |
| Warning | r | | - | o | o | o | o | o |
| WWW-Authenticate | 401 | ar | - | m | - | m | m | m |
| WWW-Authenticate | 407 | ar | - | o | - | o | o | o |

(C) 2005,2012 Technologeeks.com - All Rights Reserved




SIP and HTTP closely match one another in replies.

SIP - Replies

# SIP – Status Codes

SIP Replies contain status codes. Most align with HTTP's.

| Code | Meaning | Example Codes |
|---|---|---|
| 1xx | Informational | 100 Trying<br>180 Ringing |
| 2xx | Success | 200 OK |
| 3xx | Redirection | 301 Moved Permanently<br>302 Moved Temporarily<br>305 Use Proxy |
| 4xx | Request Failure | 401 Unauthorized<br>486 Busy Here |
| 5xx | Server Error | 500 Internal Server Error<br>501 Method Not Implemented |
| 6xx | Global Failure | 600 Busy Everywhere |

SIP Reply codes, are not only similar to HTTP's – they actually share a common subset. Still, SIP extends these codes, especially the informational ones (1xx), and adds new codes (6xx – Global Failure)

**1xx: Informational Codes:** These indicate the request is in process, but is not yet complete.

| Status Code | Reason Phrase | Meaning |
|---|---|---|
| 100 | Continue | Request is in process – received by next hop, and callee is in the process of being cotacted. |
| 180 | Ringing | Invite has reached remote party. Call Setup in progress |
| 181 | Call is Being Forwarded | Current Destination Busy – call rerouted (e.g. voicemail) |
| 182 | Queued | "Call Waiting" |
| 183 | Session Progress | Other Informatory messages concerning progress |

**2xx: Success Codes:** These indicate that the request has been processed successfully. Some Additional data may be available, pertaining to this request. Currently, only "200 OK" is defined.

| Status Code | Reason Phrase | Meaning |
|---|---|---|
| 200 | OK | Request Successful. |

(C) 2005,2012 Technologeeks.com - All Rights Reserved

**3xx: Redirection Codes:** The requested URI is not found at this location. Unlike HTTP, The Contact: header (and not Location) field is used to supply the new location.

| Status Code | Reason Phrase | Meaning |
|---|---|---|
| 300 | Multiple Choices | Multiple resources match the same URI. Attached in reply body is a list. |
| 301 | Moved Permanently | See Location: Header field for new URI. Redirect to it, and use it in the future. |
| 302 | Moved Temporarily | Temporarily moved. Next time, use original URI, but this time redirect to Location: |
| 305 | Use Proxy | This URI is only accessible via a specific proxy |
| 380 | Alternative Service | Call was not successfully, but alternatives exist. |

**4xx: Request Failure Codes:** The URI cannot be retrieved, or request cannot be fulfilled. Codes 400-416 are copied off HTTP/1.1 (Client Error). The rest, however, do not always imply the client is to blame.

| Status Code | Reason Phrase | Meaning |
|---|---|---|
| 400 | Bad Request | Request received by server was malformed |
| 401 | Unauthorized | Client requires further authorization, as per the Authorization: header |
| 402 | Payment Required | --- (reserved for future use) |
| 403 | Forbidden | Request understood, but denied |
| 404 | Not Found | Requested URI was not found on the server |
| 405 | Method not allowed | Method requested not allowed for this URI |
| 406 | Not Acceptable | Request URI does not fit client Accept: Header spec |
| 407 | Proxy Authentication Required | As per 401 (Unauthorized) but redirect to a proxy |
| 408 | Request Timeout | Request could not be fulfilled in reasonable time |
| 410 | Gone | Resource has expired and is no longer available |
| 413 | Request Entity Too Large | Client supplied entity is too large to be processed |
| 414 | Request URI Too Long | Client supplied URI in request exceeds bounds |
| 415 | Unsupported Media Type | Entity supplied by client in request is unsupported |
| 416 | Unsupported URI Scheme | URI Scheme is unrecognized or unsupported |
| 420 | Bad Extension | Server did not understand Protocol Extension |
| 421 | Extension Required | Specific Protocol Extension required |
| 423 | Interval Too Brief | Expiration time of resource is too short |
| 480 | Temporarily Not Available | Callee was contacted, but is away, or not logged in |
| 481 | Call Leg/Transaction Does not exist | Request does not match any dialong/transaction |
| 482 | Loop Detected | Forwarding loop detected |
| 483 | Too Many Hops | Max-Forwards header was 0 |
| 484 | Address Incomplete | Longer address required |
| 485 | Ambiguous | Address Specified cannot be uniquely resolved |
| 486 | Busy Here | Callee is currently busy. May suggest Retry-After, or reroute to other system. If none exist, “600” is used. |
| 487 | Request Terminated | Request terminated by a BYE or CANCEL. |

(C) 2005,2012 Technologeeks.com - All Rights Reserved

**4xx: Server Error Codes (cont.):**

| Status Code | Reason Phrase | Meaning |
|---|---|---|
| 488 | Not Acceptable Here | Callee was contacted, but could not support session parameters (bandwidth, encryption, etc..) at this end. If request cannot be supported anywhere, “606” is used instead. |
| 491 | Request Pending | Previous request is being processed. |
| 493 | Undecipherable | Encrypted request not decipherable. |

**5xx: Server Error Codes:** The requested URI cannot be retrieved, or request cannot be fulfilled, and the server is to blame. Again, 500-504 are copied off HTTP/1.1.

| Status Code | Reason Phrase | Meaning |
|---|---|---|
| 500 | Internal Server Error | Oops! |
| 501 | Not Implemented | This method is not implemented for this URI. |
| 502 | Bad Gateway | Acting as a proxy, this server received an invalid response from the upstream server |
| 503 | Service Unavailable | Server is busy, or down for maintenance (Retry-After..) |
| 504 | Gateway Timeout | Acting as a proxy, request to upstream server timed out |
| 505 | SIP Version Not Supported | SIP Version incompatible with present implementations |
| 513 | Message Too Large | SIP Message is too large to process at this server. |

SIP also adds a new error code family – “Global Failure” Errors.

**6xx: Global Failure Codes:** These are errors pertaining to the entire SIP infrastructure, not just this particular SIP server or client.

| Status Code | Reason Phrase | Meaning |
|---|---|---|
| 600 | Busy Everywhere | Callee is busy, in all registered locations. |
| 603 | Decline | The Callee actively declined. May Retry-After |
| 604 | Does Not Exist Anywhere | Server is convinced SIP User simply does not exist |
| 606 | Not Acceptable | Callee was contacted, but could not support session parameters (bandwidth, encryption, etc..) |

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SDP

# SIP and SDP

The SIP Payload is commonly “application/sdp”

The Session Description Protocol (SDP) is specified in RFC2327, and defines the setup parameters for this session.

These include:

- Session Parameters (e.g. description, addresses)
- Time Description (Date/Time of activity, repeats)
- Media Description (Transport addresses, codecs, etc..)

(C) 2005,2012 Technologeeks.com - All Rights Reserved

SDP

# SDP Headers

| Header | Header Type |
|---|---|
| v= | protocol version |
| o= | owner/creator and session identifier |
| s= | session name |
| *i=* | session information |
| u= | URI of description |
| e= | Email address |
| p= | Phone Number |
| *c=* | Connection information (not reuquired if included in all media) |
| *b=* | Bandwidth infromation |
| t= | Time session starts (one or more) |
| r= | Repeat time for session, if applicable (one or more) |
| z= | Time Zone adjustments |
| *k=* | Encryption key |
| *a=* | One or more session attribute lines |
| m= | Media name/transport address. May further specify i=,c=,b=,k=,a= per medium |

(order is mandatory, *italic* may be overridden by media Headers)

(C) 2005,2012 Technologeeks.com - All Rights Reserved

# SIP and Tunneling

- SIP defines Tunneling of messages in other messages.
- Primarily Used for:
  - Message Signing (Integrity & Authentication)
  - Message Encryption
- S/MIME used for encryption/Authentication

Example SIP Message, tunneled:

```
INVITE sip:mark@hisown.com SIP/2.0
Via: SIP/2.0/UDP kraken.hisown.com
To: My Good Friend Mark <sip:mark@hisown.com>
From: J <sip:johnny@hisown.com>
Call-ID: 24101975
CSeq: 1024 INVITE
Max-Forwards: 5
Subject: Happy 40th Birthday!
Date: Thu, 29 Dec 2016 00:00:02 GMT
Contact: <sip:jl@hisown.com>
Content-Type: multipart/signed; protocol="application/pkcs7-signature";
micalg=sha1; boundary=NextPartStartsHere
Content-Length: 620
--NextPartStartsHere
    [Tunneled SIP Message]
 --NextPartStartsHere
Content-Type: application/pkcs7-signature; name=smime.p7s
Content-Transfer-Encoding: base64
Content-Disposition: attachment; filename=smime.p7s; handling=required
[SMIME signature of content]
--NextPartStartsHere-
```

Tunneled SIP messages are encrypted similarly, with
Content-Type: application/pkcs7-mime; smime-type=enveloped-data;

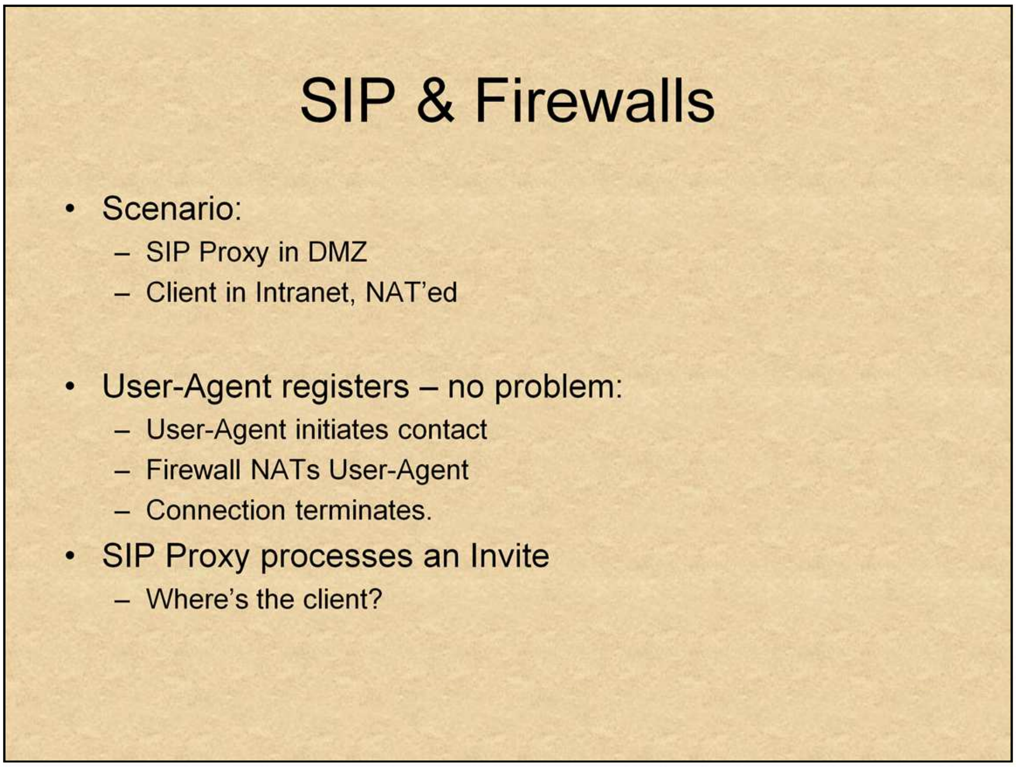
(C) 2005,2012 Technologeeks.com - All Rights Reserved

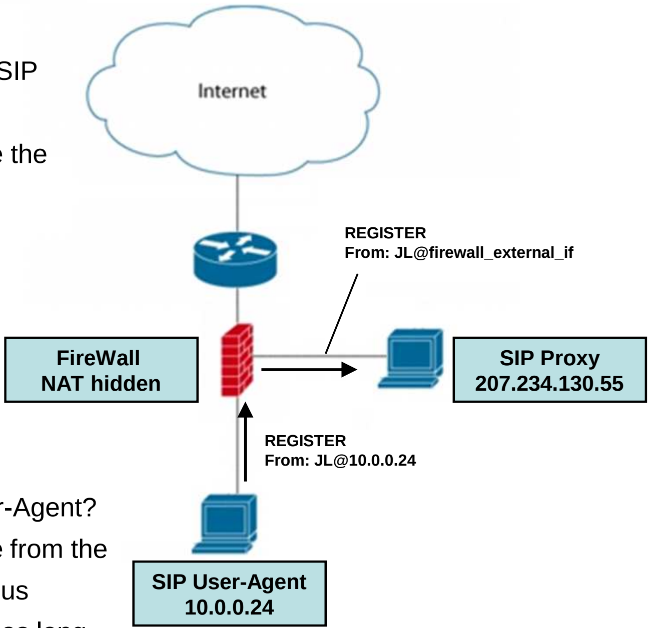
# SIP Caveats

- RFC3261 lists the following threats:
  - Registration Hijacking: usurping the user's presence
  - Rogue Servers
  - Tampering with message bodies (and call rerouting)
  - Spoofed BYEs
  - DoS

  SIP is commonly used with no authentication/encryption

(C) 2005,2012 Technologeeks.com - All Rights Reserved




<u>**Consider the following scenario:**</u>

The SIP User-Agent @ 10.0.0.24 wants to register with the SIP proxy @207.134.230.55. Since the registration request is initiated by the User-Agent, the Firewall may easily translate the REGISTER method's From: header (assuming it possesses AI or another stateful inspection mechanism.

When the SIP Proxy receives the message, it appears to Originate from the Firewall's external interface, as the NAT hides the 10.x.x.x addresses. So far so good.

However, what happens when an INVITE is sent to the User-Agent? The registration, as far as the SIP proxy is concerned, came from the Firewall, not from the end User-Agent. The SIP-Proxy will thus contact the firewall. The REGISTER connection, however, has long since been closed, and thus the User-Agent cannot be contacted. This is also a problem if there is more than one User-Agent behind the firewall.

<u>Possible solutions</u>:

- Make the internal network routable for the SIP proxy (i.e., even though it is RFC1918 addresses, enable it to be reachable from the Proxy only (Cons: potential security risk)
- Setup an additional SIP proxy inside the internet network (Cons: May be expensive)
- Use STUN or other NAT Traversal protocol.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

# SIP & Firewalls

To resolve some issues, especially NAT-related ones:

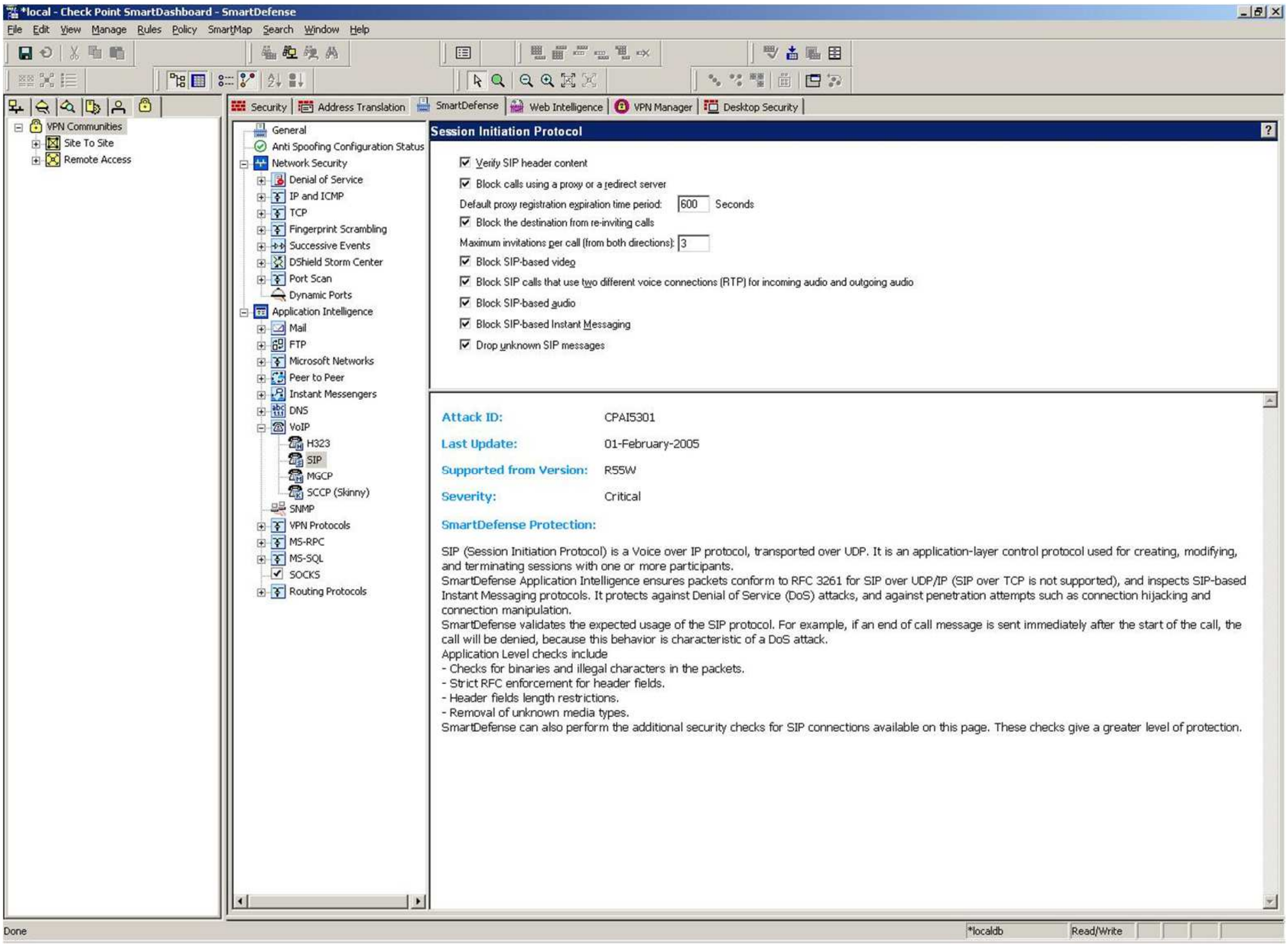
- STUN:Simple Traversal of UDP through NAT RFC3489
    - Enables User-Agent to detect its public IP Address
    - Emerging Internet Standard
    - Widely supported by many SIP devices

- ICE - Interactive Connectivity Establishment
    - Internet Draft – Status unclear
    - Builds on STUN and other protocols

(C) 2005,2012 Technologeeks.com - All Rights Reserved

# SIP Firewall capabilities

- Header Content Verification
- Blocking SIP-based video/audio/Ims
- Proxy registration timeouts
- Drop non-standard SIP methods




(C) 2005,2012 Technologeeks.com - All Rights Reserved

# Appendix

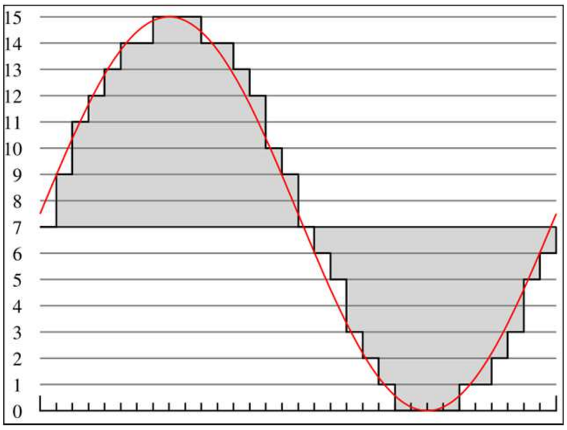
(C) 2005,2012 Technologeeks.com - All Rights Reserved

# More on Voice Codecs

- Voice is ANALOG – continuous in time & amplitude
- The Digital medium is limited to discrete samples

- To ‘digitize voice’ we therefore take discrete samples of *sample_size* bits, *sample_rate* times a second.
  - Sample size usually ranges in 4,8,16,24 or 32 bits
  - Sample rate ranges from 4,8,16,22.5 or 44.1 Khz

This illustration (taken from WikiPedia’s “PCM” entry) shows the sampling of a sinusoid signal. The continuous, analog signal is approximated by a discrete, “step” function. This is an example with a sample size of 4 bits, and a sample rate of once every clock tick.




<u>Some common sample rates and sizes:</u>

| Device | Sample Rate | Sample Size | Data/sec |
|---|---|---|---|
| Phone (u-law) | 8Khz | 8 bits | 8Khz * 8 bits = 8KB/sec = 64Kb/sec |
| AM Radio | 11.025Khz | 8 bits | 11.025 * 8 bits = 11.025KB/sec |
| FM Radio | 22.050Khz | 16 bits (per channel) | 22.025 * 16 bits * 2 (stereo) = 88.1 KB/sec |
| CD (PCM Audio) | 44.1Khz | 16 bits (per channel) | 44.1 * 16 bits * 2 (stereo) = 176.2 KB/sec |

(C) 2005,2012 Technologeeks.com - All Rights Reserved

## More on Voice Codecs

Three classes of errors result of the A-D conversion:

- **Quantization errors**: rounding to a discrete, rational value
- **Aliasing/Sampling errors**: between samples (q.v. Nyquist)
- **Aperture errors**: as a result of clock jitter

• It is the codec's responsibility to minimize all the above

Analog signals are continuous in both time and amplitude. They can be perceived as a function from time to the the signal output units, where both axes are measured in the real numbers. Digital signals, however, are of finite length – and therefore rational. As a result, digital of any resolution will still fall short of the corresponding analog.

Notice, in the example on the last page the errors are visible:

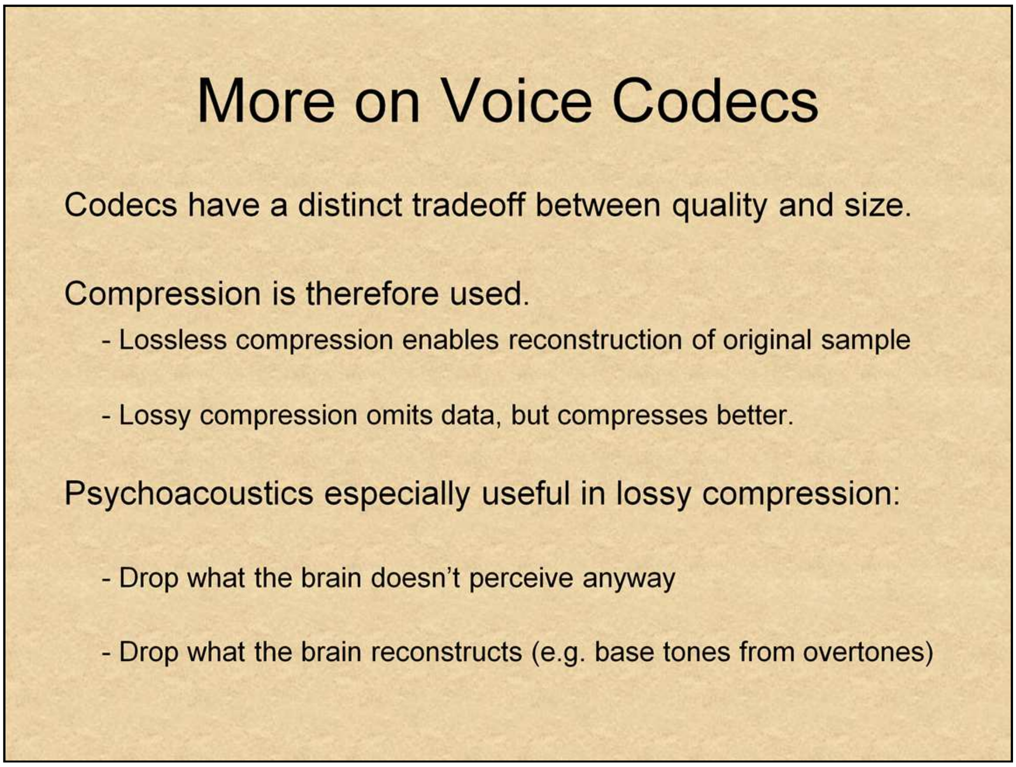
- **Quantization error:** Sample size was  is 4 bits: values from 0 through 15. As a result, the signal value is rounded to the nearest discrete value. 12.1 = 12.9 = 13 ➔ Loss of accuracy.

- **Aliasing error:** Sample rate is once every clock tick. We have no idea how the signal behaves in between samples!

(and that's assuming an accurate clock, with no Aperture errors)

Incidentally, that's also the reason why "CD quality" is at 44.1Khz. A well known theorem by Harry Nyquist (and Shannon, and a bunch of others) states that, to minimize aliasing errors one has to sample at twice the maximum analog frequency at the least. Since the maximum analog frequency is 20Khz, that means sampling rates over 40Khz are recommended (44.1Khz was chosen as it allows for filter redundancy).

(C) 2005,2012 Technologeeks.com - All Rights Reserved




As can be seen in the table showing various sample rates, more samples of greater sample size = better quality. But that also means greater bandwidth. Most codecs therefore use some form of compression.

Shannon's entropy laws put a cap on the maximum lossless compression (also called "entropy"). This is why there is a shift to lossy codecs – most notable thereof would be MPEG-2 Audio Layer 3 – commonly known as MP3.

MP3 uses some psychoacoustic traits of sound and limitations of human hearing to achieve better compression. At 128Kb/sample, it offers a roughly 1:12 ratio over PCM audio (CD). At 192Kb/sample, most humans can't distinguish its quality from that of a CD, while still allowing a 1:8 ratio.

(C) 2005,2012 Technologeeks.com - All Rights Reserved

# Suggested Reading

- http://www.packetizer.com/ - For H.323 resources
- http://www.h323forum.com/ - The "Official" site
- "IP Telephony With H.323" - Kumar et al – Wiley 0-471-39343-6
- "SIP – Understanding the Session Initiation Protocol" (2nd edition) – Johnston - Artech House 1-58053-655-7
- "Practical VoIP Security" - Porter - Syngress 1-59749-060-1
- http://www.asterisk.org/ - Home of the open source PBX

(C) 2005,2012 Technologeeks.com - All Rights Reserved

***…If you liked this course, consider…***

## Protocols:

Networking Protocols – OSI Layers 2-4:

Focusing on - Ethernet, Wi-Fi, IPv4, IPv6, TCP, UDP and SCTP

Application Protocols – OSI Layers 5-7:

Including - DNS, FTP, SMTP, IMAP/POP3, HTTP and SSL

VoIP (this course):

In depth discussion of H.323, SIP, RTP/RTCP, down to the packet level.

## Linux:

Linux Survival and Basic Skills:

Graceful introduction into the wonderful world of Linux for the non-command line oriented user. Basic skills and commands, work in shells, redirection, pipes, filters and scripting

Linux Administration:

Follow up to the Basic course, focusing on advanced subjects such as user administration, software management, network service control, performance monitoring and tuning.

Linux User Mode Programming:

Programming POSIX and UNIX APIs in Linux, including processes, threads, IPC and networking. Linux User experience required

Linux Kernel Programming:

Guided tour of the Linux Kernel, 2.6.39, focusing on design, architecture, writing device drivers (character, block), performance and network devices

## Windows:

Windows Programming:

Windows Application Development, focusing on Processes, Threads, DLLs, Memory Management, and Winsock

Windows Kernel Programming

Windows Kernel Architecture and Device Driver development – focusing on Network Device Drivers (in particular, NDIS) and the Windows Driver Model. Updated to include NDIS 6 and Winsock Kernel

## Mac OS X:

OS X and iOS Internals:

Detailed discussion on Mac OS X's internal architecture, covering aspects of performance, debugging, advanced user mode programming (threads, GCD, OpenCL), and Kernel infrastructure

**Find more courses @ http://www.technologeeks.com/courses.jl**