Volume 4, Issue 12, December 2014 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Session Initiation Protocol

B. Masthan, P. Lakshmi Devi
Department of Electronics and Communication Engineering,
Annamacharya Institute of Technology & Sciences, Rajampet, AP, India

Abstract— In recent years Voice over Internet Protocol (VoIP) has become a popular multimedia application over the Internet. At the same time critical security issues in VoIP have started to emerge. The Session Initiation Protocol (SIP) is a predominant signaling protocol for VoIP. It is used to establish, maintain and terminate VoIP calls, playing a crucial role in VoIP. The Session Initiation Protocol (SIP) provides advanced signaling and control functionality for a wide variety of multimedia services /features such as call hold and its retrieve, call waiting, Missed Call, Call Forwarding etc. This document provides examples of call flows detailing a SIP implementation of the following traditional telephony services and also presents a study over the standard VoIP protocol, Session Initiation Protocol (SIP), and how to capture the packets using the open source packet capture tool. We summarize the main protocol features and describe a range of extensions currently being discussed within the Internet Engineering Task Force.

Index Terms— VoIP, SIP

I. INTRODUCTION

SIP, like many application-layer protocols, offers several optional security services—though historically these have been used only sparingly. Even those mechanisms that the IETF mandates to implement have spotty deployment records and little prospect for interoperability, as operators and users rarely choose to enable these features. There are notable success stories, like Digest authentication as a means for users to identify themselves to services, or to a lesser extent TLS for securing hop-by-hop connections. But other security services, like true end-to-end confidentiality for personal communications, have remained elusive despite numerous attempts to render such systems as opportunistic as possible. The most frequently blamed obstacle historically has been key management: both enrolling endpoints in the keying systems confidentiality requires, and then deploying the infrastructure to discover keys in a timely manner.

But to many SIP operators, security is often seen as just another thing that can go wrong. As with other applications, users rarely clamor for privacy-enhancing features, and more significantly, even after high-profile security lapses became public scandals, users seem willing to tolerate services which lack substantial privacy protections. Also, were strong security in place, any false positives due to misconfiguration or other forms of human error would annoy users and potentially alienate customers. Accordingly, operators strive to transfer the burden of security to the service, through intermediaries, rather than pushing security information down to endpoints. This has two important consequences: first, that users typically have no way of knowing whether or not security services are in place; second, that operators are parties to any trust relationships that users establish.

II. SIP—FANTASY AND REALITY

During the design of SIP, there was significant controversy about the powers that SIP should grant to intermediaries. Drawing on personal communications architectures similar to email, SIP effectively requires an endpoint to register with an intermediary (a proxy server) in order to receive SIP calls; e.g., in order to receive calls for the "address of record" URI sip:alice@example.com, Alice's endpoint must register with the example.com SIP service. Such intermediaries implement a stateful location service for associating endpoints with an address of record when registrations are received, and then route incoming requests for that address of record to those endpoints. To perform this function, the SIP standard (RFC3261) stipulates that intermediaries may alter the incoming request in a number of respects, including:

- They may change the Request-URI from the address of record URI to a URI designating the endpoint, or any other URI the service chooses (which could for example be the address of record of a different user, in a call forwarding case)
- They may add a Via header (comparable to the "Received" header of email) which is used to route responses in the backwards direction
- They optionally add a Record-Route header which is used by endpoints to build a Route header set; Route header sets designate which intermediaries future requests in a dialog will transit

RFC3261 explicitly designated which components of a request intermediaries were permitted to change; it explicitly forbid intermediaries from altering messages bodies, for example. In practice, however, intermediaries alter messages in much broader ways than RFC3261 envisioned. For example:

- They remove any unrecognized or unauthorized SIP headers or option-tags, in order to prevent interoperability failures or the enablement of features that contradict local policy
- They change the Contact header field, in order to obscure any potential privacy sensitive information that could be leaked by the Contact URI (some devices build the Contact URI using the IP address of the endpoint), and also to prevent endpoints from sending traffic to one another directly when that contradicts local policy
- They change the Call-ID, as the Call-ID is created by some legacy devices through concatenating a string containing the endpoint's IP address, with its potential privacy sensitivities
- They remove Via headers in order to provide further "topology hiding," that is, to prevent leaking to untrusted endpoints details about prior intermediaries in order to keep these secret, either to protect business arrangements or to obscure targets from potential hackers
- They remove Record-Route or Route headers for similar reasons, and moreover to prevent endpoints from routing future requests in ways that contradict local policy
- Most significantly, they change the contents of SIP message bodies, particularly the Session Description Protocol (SDP) bodies carried by the INVITE, 183 and 200 messages, in order to alter IP address, ports and codecs associated with proposed media streams; most commonly to facilitate NAT traversal, conference or transfer, or transcoding

This mismatch between the intended powers of intermediaries in the standard and the actual powers that intermediaries exercise in the field has significant implications for SIP security. Any entity that can change the IP address and ports of SDP can direct the media of real-time communications wherever they would like, which enables various denial of service or impersonation attacks. It also permits any intermediary to send media through a simple relay for recording or distribution. Entities that can modify SDP can moreover insert themselves as men-in-the-middle for any key exchange conducted through SDP to secure media via SRTP. In order to detect an attack where a man-in-the-middle modified SDP, RFC3261 provided services for body-level security with S/MIME signatures. This requires a PKI and a key discovery mechanism, neither of which are trivial to provide for a protocol like SIP. Even if keying were resolved, however, operators deploying SIP so frequently relied on intermediaries modifying SDP that signing the body of SIP requests was simply infeasible. The attack of a man-in-the-middle could not be distinguished from the legitimate behavior of an operator's intermediary.

III. THE SIP ARCHITECTURE

SIP is an application-layer signaling protocol for handling Multimedia sessions over the Internet. In a typical SIP-based Network infrastructure, the following network elements are Involved:

- **User Agents**: user agents (UAs) act on behalf of an end user terminal. A user agent client (UAC) is responsible to create requests and a user agent server (UAS) processes and responds to each request generated by a UAC.
- **Registrar**: UAs contact registrar servers to announce their presence in the network. The SIP registrar server is a database containing locations as well as user preferences as indicated by the UAs.
- **Proxy**: A proxy server receives a request and forwards it towards the current location of the callee — either directly to the callee or to another server that might be better informed about the actual location of the callee.
- **Redirect**: A redirect server receives a request and informs the caller's UA about the next hop server. The caller's UA then contacts the next hop server directly.

Various types of text based messages have been introduced in SIP following the HTTP message structure [5]. SIP messages must also identify the requested resource, which corresponds to a unique address. The SIP address (SIP-URI) is aligned with the general form of the HTTP addressing scheme, which is: "address scheme: resource." As a result, a user is identified through a SIP URI in the form of sip:user@domain. As an example, the URI sip:zintan@real.com is a valid SIP address. This address can be resolved by a SIP proxy that is responsible for the user's domain. The first step for a user to use a SIP-based service is to identify his/her actual location in terms of an IP address. Consequently, the user needs to register the combination of his/her SIP address and current IP address at the SIP registrar responsible for his domain. This registration procedure is depicted in Figure 1

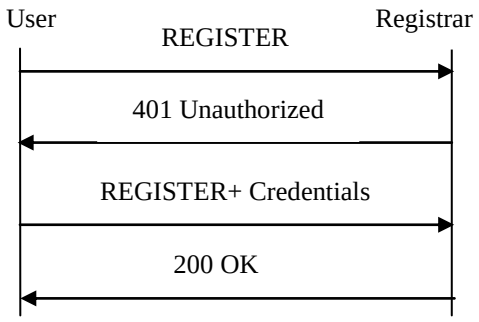

Figure 1: Registration Procedure

When inviting a user to participate to a call (callee), the calling party (caller) sends a SIP INVITE to the corresponding SIP proxy, which checks in the registrar's database or in the Domain Name System (DNS), the location of the callee and forwards the invitation to the callee. The latter can either accept or reject the invitation. During this message exchange, both the caller and the callee exchange the addresses/ports at which they would like to receive the media as well as the type of media (i.e., video, voice) they can accept. After finalizing the session establishment, the end systems can exchange media data directly without the involvement of any SIP proxy. This procedure is depicted in Figure 2.

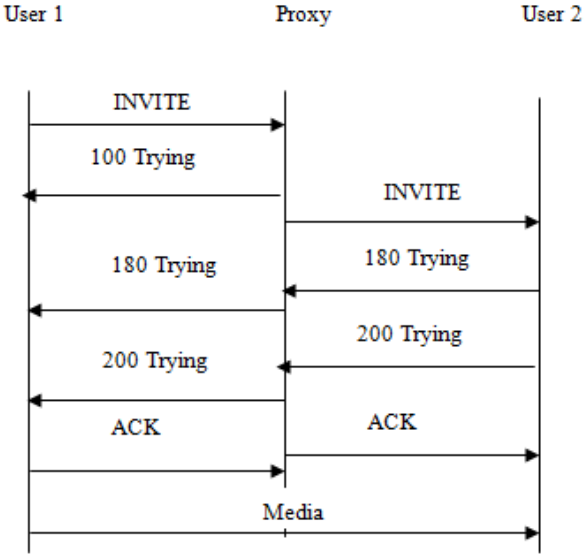

Figure: 2 Calling a User in SIP

However, under certain circumstances the aforementioned procedure is not feasible because the corresponding proxy may be temporarily unavailable (e.g., through overload, or because of software update). Under such situations the mediation of a Redirect server is required in order to inform the caller (user 1) on possible alternative locations to reach the requested URI. As soon as the caller receives this information, he/she generates a new request towards one of the alternative locations. This procedure is depicted in Figure 3.

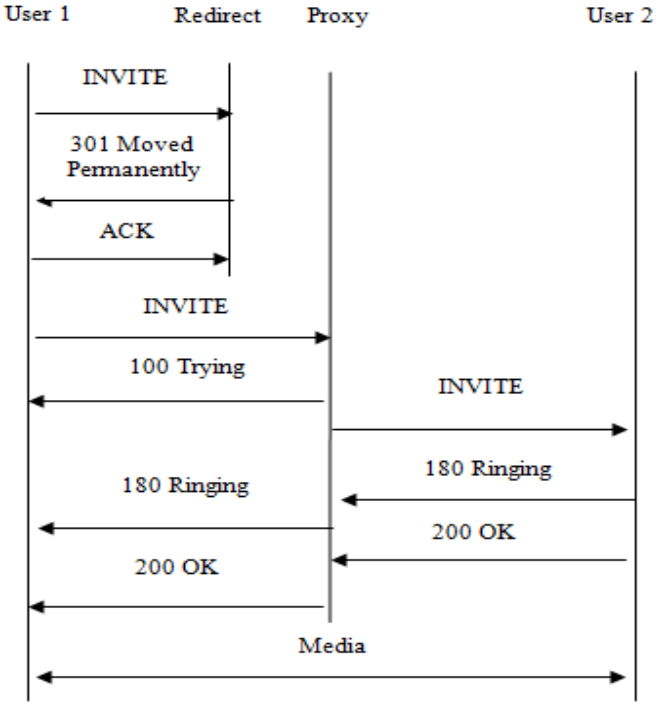

Figure: 3 SIP Call utilizing a Redirect proxy.

IV. RESULTS AND SUMMARY

This project is tested by using the soft phones and Asterisk Server. This Asterisk Server acts as sip proxy server. The Sip proxy server is mainly used to forward the sip messages between the endpoints. Server receives the request from the UAC and then forwards that request to UAS, and also server receives response from the UAS forwards it to UAC. For testing the sip services we should create the extension numbers and then assign the extension numbers to phones. Register user with server by giving username and password.

- **Basic Call:** For testing the Basic Call/end to end call, dial number using dialpad and verify dialed numbers should be displayed and the following call flow is shown in figure 4.The wireshark tool is used capturing the traces of Basic Call and is shown in figure 5.

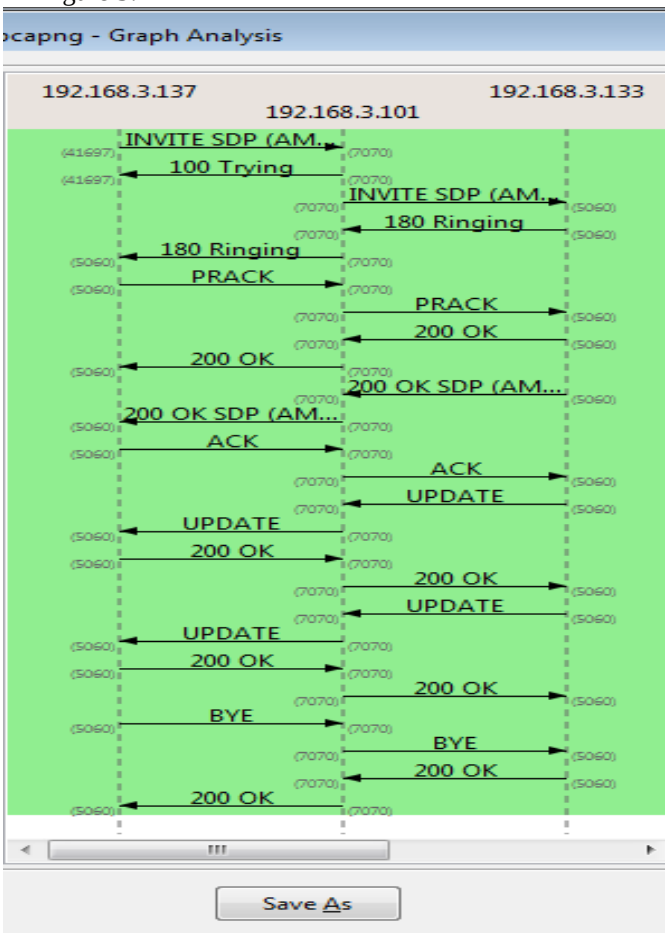

Figure 4: Basic call flow

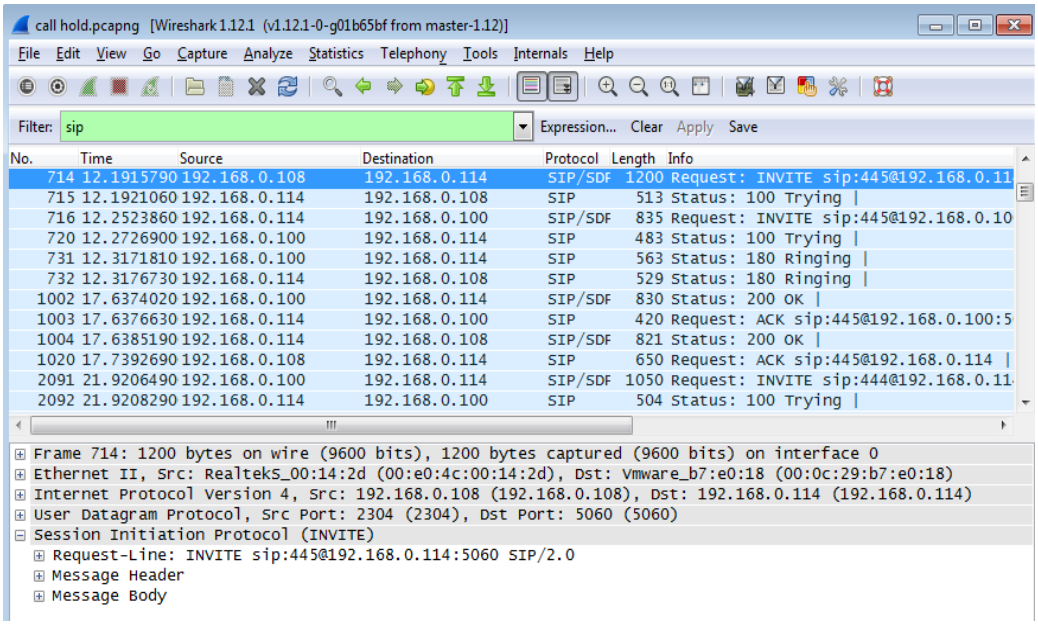

Figure 5: Basic call Traces

- **Call Hold feature:** In this scenario, User A calls User B, then User A places the call on hold. User A then takes the call off hold, then User A hangs up the call. Note that hold is unidirectional in nature. However, a UA that places the other party on hold will generally also stop sending media, resulting in no media exchange between the UAs. Older UAs may set the connection address to 0.0.0.0 when initiating hold. However, this behavior has been deprecated in favor or using a=inactive SDP attribute if no media is sent, or the a=sendonly attribute if media is still sent. The call flow is shown in figure 6 and a trace is shown in figure 7.

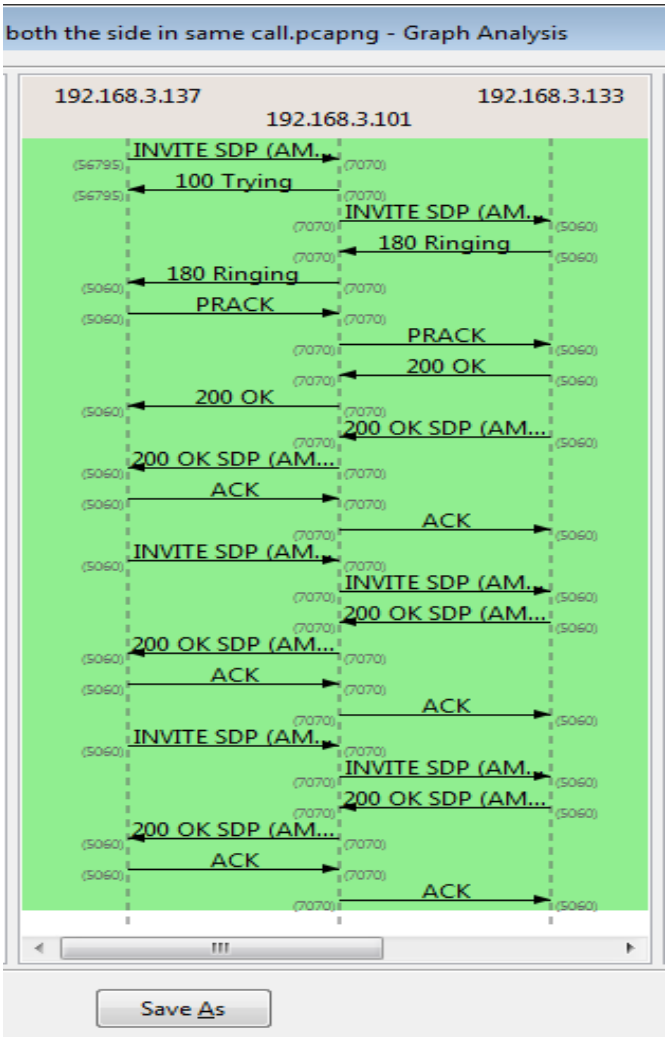

Figure 6: Call flow of Call Hold

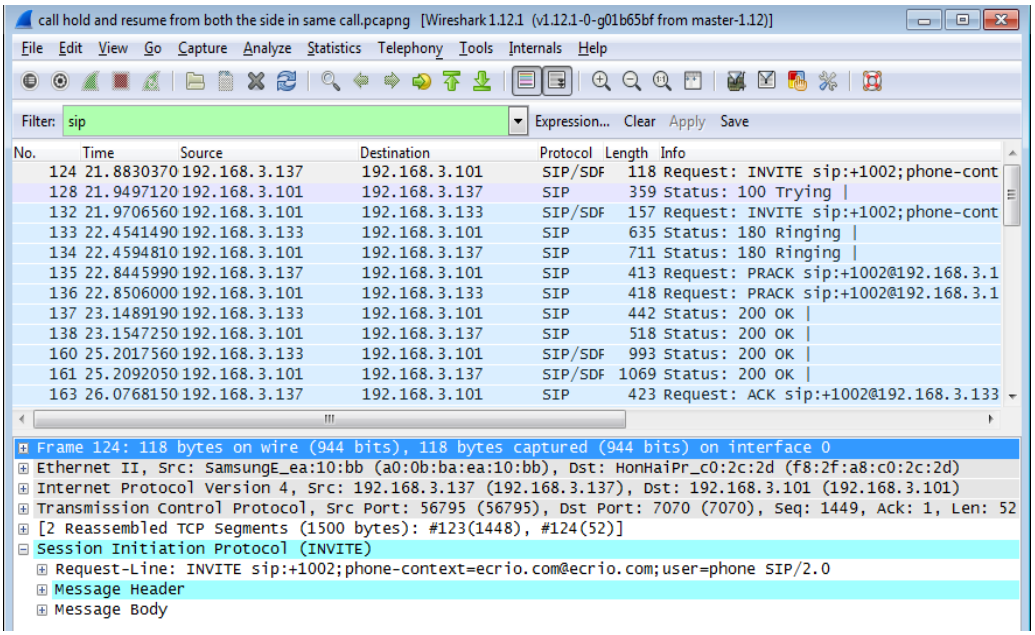

Figure 7: Traces of Call Hold

- **Missed Call/Cancel by the Originator:** In this scenario, User A calls User B by dialing the User B number from the dialpad of User A phone. User B starts ringing when it receives request from the User A. User A wants to cancel the request, since he sends the CANCEL request to the User B and it should be accept by the User B. The Call flow of Missed call is shown in figure 8 and the traces of missed call is shown in figure9.

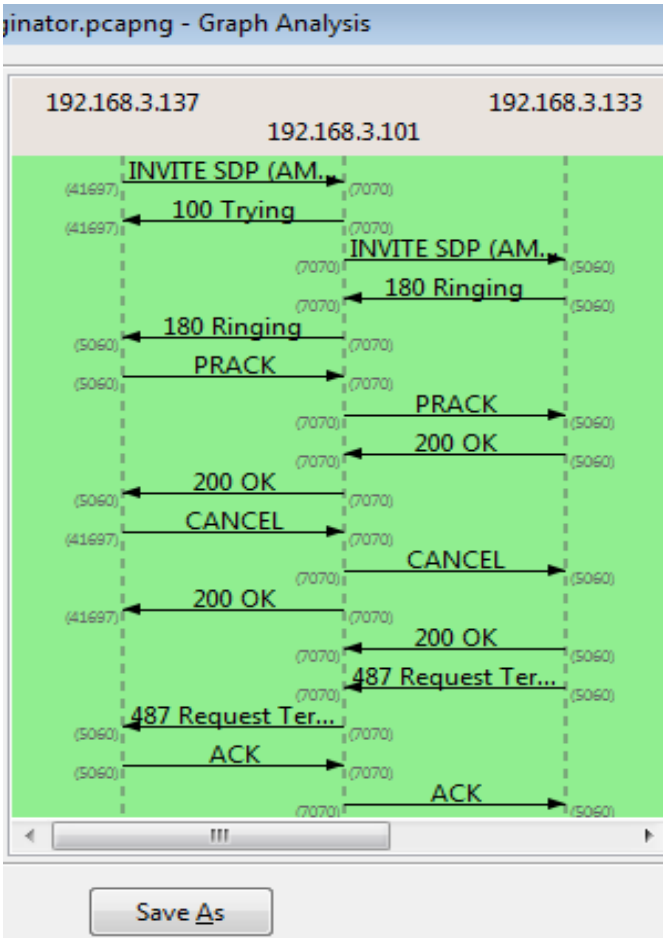

Figure 8: Call flow of Missed Call

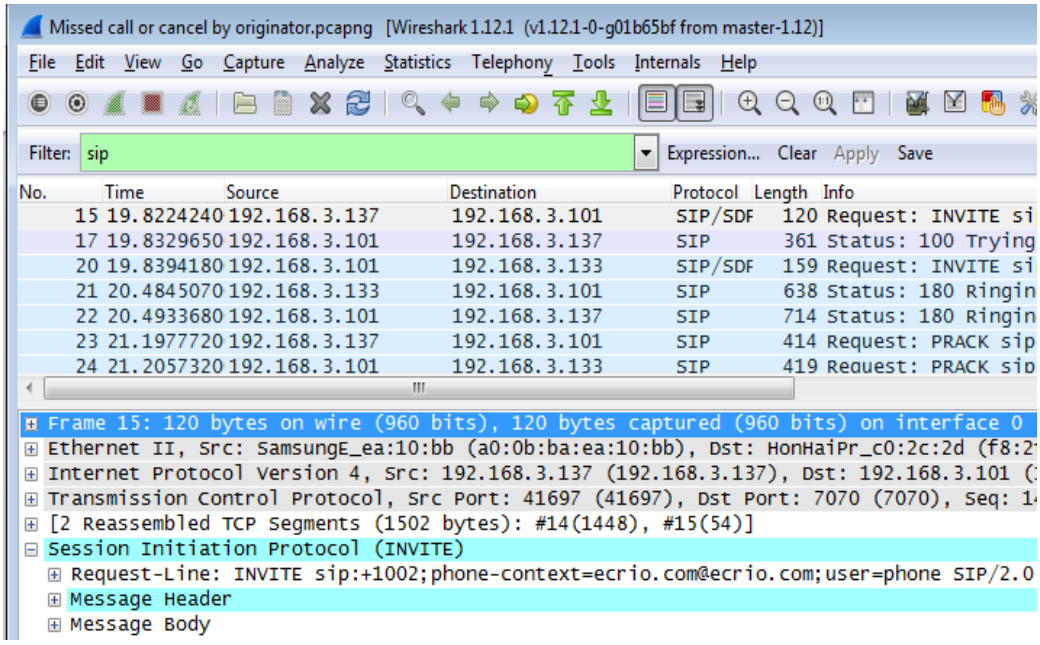

Figure 9: Traces of Missed Call

V. CONCLUSION

Session Initiation Protocol (SIP) has grown a lot since it first became an IETF standard in 1999. SIP was originally intended purely for video and audio and now has grown as the control protocol for many interactive services, particularly in the peer-to-peer realm. SIP and the standards surrounding SIP provide the standards-based mechanisms to look up, negotiate, and manage connections to peers on any network over any other protocol. I hope by now you have got a basic idea of what SIP is and what it does. You should be able to recognize the major components in a SIP scenario and how different messages are exchanged to establish and terminate sessions.

REFERENCES

[1] http://www.rfc-base.org/txt/rfc-3261.txt
[2] J. Rosenberg, et al., "SIP: session initiation protocol," 2001.
[3] R. Canetti and H. Krawczyk, "Analysis of key-exchange protocols and their use for building secure channels," Advances in Cryptology—EUROCRYPT 2001, pp. 453-474, 2001.
[4] J. Davidson, et al., Voice over IP fundamentals: Cisco Systems, 2006.
[5] D. E. Denning and G. M. Sacco, "Timestamps in key distribution protocols," Communications of the ACM, vol. 24, pp. 533-536, 1981.
[6] W. Diffie and M. Hellman, "New directions in cryptography," Information Theory, IEEE Transactions on, vol. 22, pp. 644-654, 1976.
[7] A. Durlanik and I. Sogukpinar, "SIP authentication scheme using ECDH," World Enformatika socity Transaction on Engineering computing and technology, vol. 8, pp. 350-353, 2005.
[8] J. Franks, et al., "HTTP Authentication: Basic and Digest Access Authentication," 1999.
[9] N. Koblitz, "Elliptic curve cryptosystems," Mathematics of computation, vol. 48, pp. 203-209, 1987.
[10] A. J. Menezes, et al., Handbook of applied cryptography: CRC, 1997.
[11] S. Salsano, et al., "SIP security issues: the SIP authentication procedure and its processing load," Network, IEEE, vol. 16, pp. 38-44, 2002.
[12] J. L. Tsai, "Efficient nonce-based authentication scheme for session initiation protocol," International Journal of Network Security, vol. 8, pp. 312-6, 2009.
[13] L. Wu, et al., "A new provably secure authentication and key agreement protocol for SIP using ECC," Computer Standards & Interfaces, vol. 31, pp. 286-291, 2009.
[14] C. C. Yang, et al., "Secure authentication scheme for session initiation protocol," Computers & Security, vol. 24, pp. 381-386, 2005.

ABOUT AUTHOR

B. Masthan born in Nellore, A.P, India in 1990. He received B.Tech Degree in Electronics &Communication Engg. From J.N.T.University, Anantapur, India. Presently he is pursuing M.Tech (DECS) from Annamacharya Institute of Technology & Sciences, Rajampet, A.P., India. His research interests include Session Initiation Protocol.

Mrs Lakshmi Devi is graduated from SVU College of engineering Thirupathi and obtained Masters Degree from JNTU Anantapur, Andhra Pradesh. Currently she is working as an Associate Professor at AITS, Rajampet for the last 14 years. Her research interests includes Signal Processing, Image Processing, Evolutionary Algorithms etc.

