Real-world SIP Interoperability: Still an Elusive Quest

Archana Rao and Henning Schulzrinne
Columbia University

Abstract—**With more than a decade of development led by the IETF, and a plethora of devices and software systems speaking its dialect, SIP together with its related standards has grown in size and scale, raising concerns over interoperability. In this paper, we explore SIP interoperability (or lack thereof) by proposing systematic methodologies for identifying and analyzing the basic-level protocol interoperability issues that plague SIP usage in the real-world. We also dissect and describe a few of the commonly observed SIP interoperability issues and their implications. Our test results clearly indicate that even the basic-level of SIP interoperability is far from ideal.**

I. Introduction

With the Session Initiation Protocol (SIP) [1] becoming the de-facto signaling protocol for real-time communications on the Internet, the number of implementations of the protocol have increased dramatically over the last few years. Fueled by a rapid adaptation by a variety of domains and applications, SIP has grown from being a single protocol to a system of protocols with nearly 150 RFCs [2] and a few hundred active Internet-Drafts defining the various aspects of SIP-based real-time communications. While these developments create an ideal platform for engineers and researchers to further enhance the body of knowledge and for end-users to reap the benefits of technological advancements and unification, such benefits come at a cost - the foremost being the issue of interoperability.

The SIP interoperability issues arise due to a variety of reasons. First, owing to the complexity, multitude and continuous evolution of SIP-related RFCs and Internet-Drafts, the implementers are faced with a non-trivial challenge of choosing the right set of SIP-related protocols, sorting out the nuances at the boundaries, and keeping track of changes throughout the evolution of specifications from Internet-Drafts to RFCs. Efforts like Hitchhiker's Guide to SIP [2] help to a certain extent, but are by no means sufficient. Second, unlike many other standards from the ISO and the ITU, SIP has neither a proposed architecture nor a set of profiles, making it difficult to establish standards-compliance. Third, the vendors extend and modify their SIP implementations well beyond the protocol specifications in order to deploy their products in variety of non-standard environments. Finally, one cannot rule out the intentional non-interoperability induced to prevent the use of certain third party products.

There have been several efforts to address SIP interoperability issues. The most notable being the SIP Interoperability (SIPit) events [3], organized by the SIP Forum as a bi-annual week-long gathering of SIP implementers to identify, discuss and sort out interoperability issues. Similar attempts have been tried by other organizations including the University of New Hampshire InterOperability Laboratory [4]. Realizing that some of the advanced-level SIP features such as call parking, call transfers do not work as expected in a multi-vendor environment, the IETF Basic Level of Interoperability for SIP Services (BLISS) working group [5] was formed in July 2007. Despite these efforts, SIP interoperability has not attracted much attention in the engineering and research environments, and has largely remained a pursuit taken out of one's own interest.

Due to lack of compelling reasons that could force SIP vendors towards achieving the level of device interoperability that exists in the PSTN world, the trends have been such that even a basic level of interoperability amongst the SIP devices, is no longer trivial. It would be prudent to ask ourselves – ***can we take any SIP phone and make a VoIP call to any other SIP end-client, through any SIP service provider?*** The answers are not always in the affirmative. The chances of multi vendor equipments working seamlessly in all the environments are extremely rare.

In this paper, we propose systematic approaches to identify, analyze and solve the real-world SIP interoperability issues. The remainder of the paper is organized as follows. Section II describes our methodology and infrastructure to identify and study the SIP interoperability issues. Section III analyzes and categorizes a few of the commonly observed SIP interoperability issues. Section IV proposes measures to develop effective solutions against real-world SIP interoperability problems. Finally, section V concludes the paper.

Manuscript received November 23, 2007. This work was supported in part by the National Science Foundation under grant # 0551694.

Archana Rao is with the Electrical Engineering Department, Columbia University, New York, NY 10027 USA (e-mail: ahr2114@columbia.edu).

Henning Schulzrinne is with the Computer Science Department, Columbia University, New York, NY 10027 USA (e-mail: hgs@cs.columbia.edu).

II. A Systematic Approach To Interoperability

A. Framework for Identifying Interoperability Issues

In order to perform a systematic study, the first step is to identify what would constitute the basic-level of SIP interoperability. SIP Basic Call Flow Examples [6] describes a set of SIP call flow scenarios covering SIP registration and session establishments, which establish the minimum set of functionality present in a SIP communication network. Since these call flows represent the working group reviewed scenarios, intended as a companion to the SIP protocol for implementers, designers and researchers, one can use them as the basis vector for SIP interoperability studies. This best practices document, describes a total of 5 SIP registration and 11 SIP session establishment scenarios.

The SIP Public Switched Telephone Network (PSTN) Call Flows [7] and the Session Description Protocol (SDP) Offer/ Answer examples [8] are two further references that extend this basic set of call flow scenarios, with the former describing 18 examples of SIP-PSTN interworking and the latter describing 13 examples of codec negotiation and selection, and addition and deletion of media streams. We propose this set of 47 call flows to constitute the minimum set of scenarios that should seamlessly work with any SIP device in any SIP IP communication environment.

With the basic set of test vectors identified, the next task is to run these tests in the real-world SIP infrastructure setups, using an indicative sample of SIP end-clients and SIP servers that are widely deployed in residential, commercial, educational and enterprise environments. Such an approach would not only enable us to identify the real-world interoperability problems that end-users often encounter, but also build a knowledge system that can inform the users about the existing issues. Realizing the interoperability study on this scale needs a large-scale, distributed and configurable VoIP testbed – and we use the Columbia VoIP Testbed (described in the following section) for this purpose.

B. Columbia VoIP Testbed

Columbia VoIP Testbed is an NSF-supported [9] research platform that provides a VoIP infrastructure for experimentation, analysis, testing, prototyping and deployment of SIP components in a variety of environments. Primarily made up of four sets of components namely SIP servers, SIP end-clients, network devices and support infrastructure, the testbed is also connected to Purdue University and University of North Texas through a Virtual Private Network (VPN) setup. The interconnection architecture of the Columbia VoIP testbed is as depicted in Figure 1.

SIP Server Farm is the core of the architecture comprising of SIP registrar, redirect and proxy servers. It comprises of *five* of the most widely used publicly available SIP servers, running on *three* different platforms (Microsoft Windows XP, Linux Fedora Core 6 and Sun Solaris 10) and providing connectivity through *three* different networks (VPN, the Internet, and the PSTN). The testbed has more than 20 **SIP end-clients** of varying capabilities (hard and soft phones, wireless phones and video phones) from different vendors, integrated and tested in the setup. **Network devices and support infrastructure** make the testbed setup configurable and help in realizing various logical topologies over the existing physical resources. These include a Cisco 7801 VPN router, wireless access points, NAT devices, Ethernet switches and hubs, DNS and DHCP servers, wireline and wireless Internet services, among others.

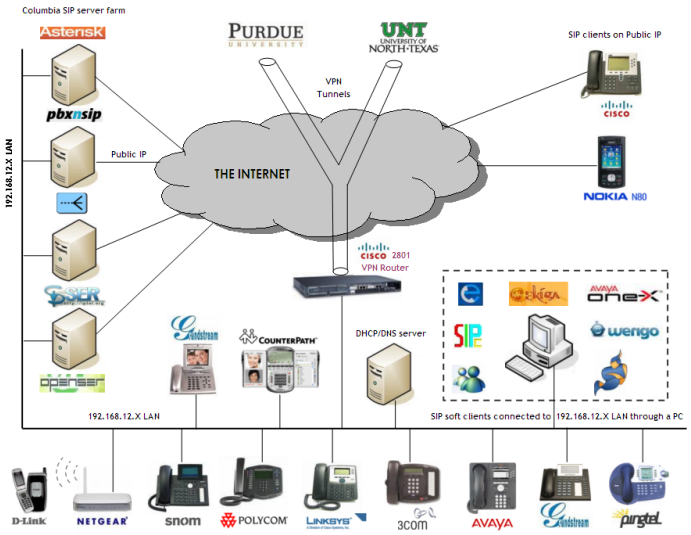

Fig.1. Columbia VoIP Testbed – interconnection diagram

III. Real-world Interoperability Failures

This section describes a few of the commonly observed interoperability issues, most of which were identified during our experiments on the testbed infrastructure, while some were reported by others and verified on the testbed. Each issue is organized as a description, implication pair and described without any reference to the vendor products. In order to facilitate a systematic treatment, we organize these issues into five broad categories.

In this section, we use the term "specification" to refer to the umbrella of standards that describe the behavior in discussion, including the RFCs, Internet-Drafts and any other relevant standards document.

A. Lack of Clarity in the Specification

Interoperability issues could crop up, when the specification is silent on a specific aspect, and one such instance is discussed below.

1. Use of different formats for authentication name

SIP provides a stateless, challenge-based mechanism for authentication that is based on digest authentication in HTTP. When using digest authentication, a User Agent Client (UAC) trying to establish a session with a User Agent Server (UAS) or trying to register with a registrar, may be challenged by its

proxy or the registrar to provide its credentials. The UAC then resends its request along with the authentication information in the *Authorization* header. We have identified that UACs follow different formats for the authentication username field, while composing this header. Three such variations are as shown.

Authorization: Digest ***username="user",*** *realm="domain", nonce="xxx", uri="sip:proxy.provider.com", response="yyy", algorithm=MD5*

Authorization: Digest ***username="sip:user@domain",*** *realm="domain", nonce="xxx", uri="sip: proxy.provider.com", response="yyy", algorithm=MD5*

Authorization: Digest ***username="user@domain",*** *realm="domain", nonce="xxx", uri="sip: proxy.provider.com", response="yyy", algorithm=MD5*

We have observed that some registrars and proxies do not accept all three variations of the authentication username, resulting in failed registration and/or failed session initiation. This issue occurs when a UAC cannot be configured to provide the authentication username in the format that is accepted by the registrar or the proxy. We have observed in our experiments that all of the registration failures (barring cases where the end-users have supplied wrong credentials) could be attributed to this issue.

B. Implementation of an Older Specification

This category encompasses issues that arise from the SIP devices implementing deprecated RFCs and/or expired Internet-Drafts.

1. Use of incompatible payload type for RTP codecs

The RTP payload type 2 was assigned to G.721 in the RTP Profile for Audio and Video Conferences with Minimum Control [10] and later assigned to its successor G.726-32. But when a newer RFC [11] deprecated the older one [10], the RTP static payload type 2 was marked as reserved. Despite this clarity in the specification, we have seen a proxy server using static type 2 for G.726.

Such an issue can result in a codec negotiation failure, despite both the SIP end-clients supporting common formats.

C. Incomplete Implementation of the Specification

This section discusses two issues arising from SIP devices that are unable to handle advanced cases of the specification.

1. Different levels of support for DNS queries

SIP: Locating SIP Servers [12] proposes DNS procedures to allow a client to resolve a SIP Uniform Resource Identifier (URI) to the IP address, port, and transport protocol of the next hop to contact. A high percentage of UAs support only DNS A query, few others have DNS SRV as an advanced feature (which needs to be configured manually), and very few have all of A, SRV and NAPTR queries by default. This difference in support for DNS queries could lead to situations where two UAs select different transport protocols (e.g., TCP rather than UDP) for the same next hop server (registrars or proxies).

While this should have no problem in theory, in practice we have seen that SIP signaling over TCP generally failing in a multi-vendor environment.

2. Session establishment with multiple proxy authentications

Despite being well specified in [6], session establishment with multiple proxy authentications is not correctly supported by a majority of SIP UAs. Referring to the call flow in Figure 2, we have seen scenarios, where signaling does not progress beyond F9, even though UAC has valid credentials in both the domains. The scenario becomes completely unworkable if the UAC has different set of credentials in the two domains. No SIP end-client that we have experimented with supports multiple sets of username and password credentials.

Such an issue leads to session establishment failures, when the SIP INVITE has to traverse untrusted domains.

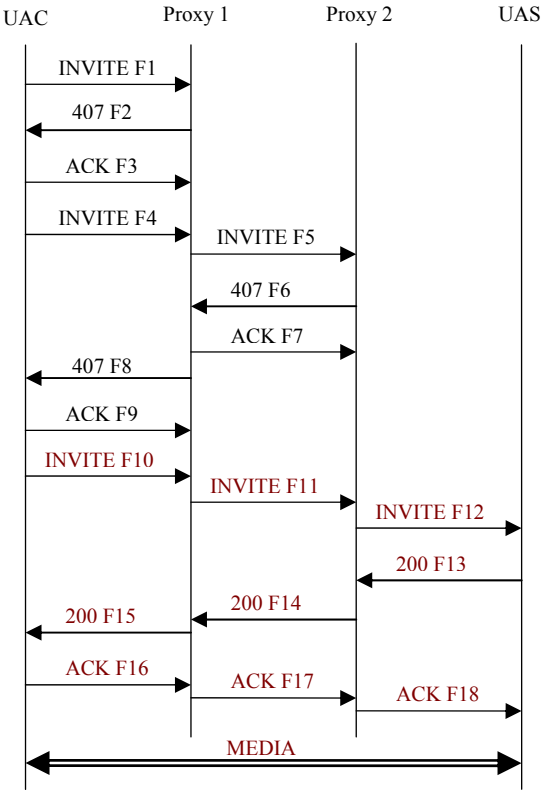

Fig.2. SIP Call flow for session with multiple proxy authentication

D. Incorrect Implementation of the Specification

1. Lack of support for non-symmetric signaling

The Contact header field provides a SIP URI that can be used to contact the UA for subsequent requests. Specifically, a UAS, in its responses, adds a Contact header field that

indicates the address where it would like to be contacted for subsequent requests in the dialog (which includes the ACK for a 2XX response in case of an INVITE). We have observed that a few proxy servers send ACKs back to the source port from where the "180 Ringing" and "200 OK" responses were received, rather than to the port in the contact header of UAS.

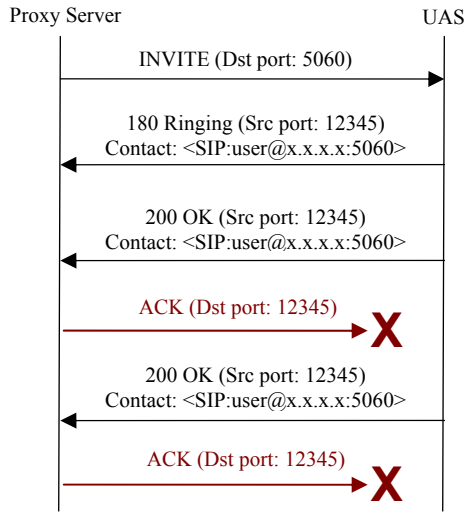

Fig.3. SIP Call flow showing the ACKs being sent to wrong port

Such a behavior would result in multiple retransmissions and signaling failure. Call flow in Figure 3 shows part of the signaling between UAS and its immediate neighboring proxy, where ACK is wrongly sent to port 12345, while UAS is expecting it at port 5060.

2. Unsuccessful cancellation of registration

We have seen at least two scenarios of unsuccessful cancellation of registrations. The first one, when a UA does not care to resend the registration cancellation request upon getting '401 Unauthorized' response from the registrar. The second one concerns the use of '*' in the contact header, which requests the registrar to remove all contract bindings for the user. We have observed that not all registrars honor such requests for registration cancellation.

Such an unsuccessful cancellation of registrations may lead to undesirable behavior during signaling, due to incorrect Address of Record (AOR) details.

3. Media failure in case of codec reordering

While it should be possible for a UAC and a UAS to use different audio codecs in the same VoIP call, we have seen that some UAs are unable to handle this situation. Figure 4 shows an SDP offer and answer exchange between a UAC and a UAS, where a codec preference specified in the offer is changed during the answer by codec reordering.

Once this codec negotiation is complete, the UAC sends its audio packets in the PCMU format while the UAS in G.729. But no audio could be heard at the UAC.

[Offer]	**[Answer]**
v=0 o=UA1 aa bb IN IP4 x.x.x.x s= SIP Call t=0 0 m=audio 22238 RTP/AVP 0 8 18 101 c=IN IP4 xx.xx.xx.xx a=rtpmap:0 PCMU/8000 a=rtpmap:8 PCMA/8000 a=rtpmap:18 G729/0 a=fmtp:18 annexb=no a=rtpmap:101 telephone-event /8000 a=fmtp:101 0-15 a=sendrecv	v=0 o=UA2 cc dd IN IP4 y.y.y.y s= session c=IN IP4 y.y.y.y t=0 0 m=audio17088 RTP/AVP 18 0 8 101 a=rtpmap:18 G729/8000 a=fmtp:18 annexb=no a=rtpmap:0 PCMU/8000 a=rtpmap:8 PCMA/8000 a=rtpmap:101 telephone-event /8000 a=fmtp:101 0-16 a=silenceSupp:off - - -

Fig.4. SDP offer/answer, with codec reordering

Such a behavior by UAs could result in no audio being heard by one or both end users, despite successful signaling.

E. Failure against Robustness Tests

We evaluated the performances of SIP end-clients against two of the widely used robustness tests. The SIP Torture Test Messages [13] contains a set of 49 SIP messages, developed and refined at SIPit events. These tests primarily focus on areas that have caused interoperability problems or that have particularly unfavorable characteristics if handled improperly. The second was the PROTOS SIP Test Suite [14] developed by the University of Oulu, which contains more than 4500 SIP INVITE requests. Both these test suites are designed to exercise and torture the SIP parser and the application above the SIP implementation.

Our testing showed that about 50% of the SIP end-clients in our testbed passed all the tests successfully. Others demonstrated varying levels of undesired behavior including software crash, unresponsive hardware, and a dramatic increase in CPU consumption. While a majority of the SIP end-clients showed failures against one particular set of tests e.g., incorrect values for Content-length field, few of them performed badly against more than one category.

IV. Future Direction

SIP interoperability has been an elusive quest so far. We propose a set of measures that we believe would provide effective solutions, in conjunction with the current efforts. First, we strongly recommend establishing designated liaisons for each vendor that have the ability to make the software development staff aware of the issues. Second, the vendor should be encouraged to publish interoperability reports so that the consumers are aware of the interoperability problems before purchasing equipments and software. Third, a set of self-certification tests should be provided, possibly supported by a remotely accessible test rig that allows implementers to test their user agents.

V. Conclusion

We propose a systematic approach to identify and analyze the SIP interoperability issues in the real-world. Our experiments have clearly indicated that SIP interoperability is nowhere close to 100% even for basic-level call flows. It is unfortunate to see that SIP end-clients fail to pass the robustness tests, even years after such tests being around. We believe that achieving seamless interoperability at the basic-level is crucial and should not be left to the discretion of the vendors or as a matter of self-interest. It is also our hope that people feel encouraged to use Columbia VoIP testbed to identify, demonstrate and fix real-world issues.

Acknowledgment

We would like to thank the members of the Internet Real-time (IRT) Laboratory, Columbia University for their valuable inputs during the interoperability testing and also for the review comments. We would also wish to thank Phil Karn for reporting an issue that led to our investigation on the lack of support for non-symmetric signaling.

References

[1] J. Rosenberg, H. Schulzrinne, G. Camarillo, A Johnston, J Peterson, R. Sparks, M. Handley, and E. Schooler, "SIP: Session Initiation Protocol," RFC 3261, Internet Engineering Task Force, June 2002.

[2] J. Rosenberg, "A Hitchhiker's Guide to Session Initiation Protocol," draft-ietf-sip-hitchhikers-guide-04 (work in progress), November 2007.

[3] SIP Interoperability Tests (SIPit), https://www.sipit.net

[4] University of New Hampshire InterOperability Laboratory (IOL), http://www.iol.unh.edu

[5] Basic Level of Interoperability for SIP Services (BLISS), http://www.ietf-bliss.org

[6] A Johnston, S. Donovan, R. Sparks, C. Cunningham, and K. Summers, "Session Initiation Protocol Basic Call Flow Examples," RFC 3665, Internet Engineering Task Force, December 2003.

[7] A Johnston, S. Donovan, R. Sparks, C. Cunningham, and K. Summers, "Session Initiation Protocol Public Switched Telephone Network Call Flows," RFC 3666, Internet Engineering Task Force, December 2003.

[8] A Johnston, S. Donovan, R. Sparks, "Session Description Protocol Offer/Answer Examples," RFC 4317, Internet Engineering Task Force, December 2005.

[9] NSF CRI Testbed, http://nsl.unt.edu/CRI

[10] H. Schulzrinne, "RTP Profile for Audio and Video Conferences with Minimum Control," RFC 1890, Internet Engineering Task Force, January 1996.

[11] H. Schulzrinne, S. Casner, "RTP Profile for Audio and Video Conferences with Minimum Control," RFC 3551, Internet Engineering Task Force, July 2003.

[12] J. Rosenberg, H. Schulzrinne, "Session Initiation Protocol (SIP): Locating SIP Servers," RFC 3263, Internet Engineering Task Force, June 2002.

[13] R. Sparks, A. Hawrylyshen, A. Johnston, J. Rosenberg, H. Schulzrinne, "Session Initiation Protocol (SIP) Torture Test Messages," RFC 4475, Internet Engineering Task Force, May 2006.

[14] PROTOS Security Testing of Protocol Implementations, http://www.ee.oulu.fi/research/ouspg/protos/testing/c07/sip