Understanding Session Border Controllers

FRAFOS GmbH

Table of Contents

1 Introduction

Over the past 10 years the Session Initiation Protocol (SIP) has moved from the toy of researchers and academics to the de-facto standard for telephony and multimedia services in mobile and fixed networks.

Probably one of the most emotionally fraught discussions in the context of SIP was whether Session Border Controllers (SBC) are good or evil.

SIP was designed with the vision of revolutionizing the way communication services are developed, deployed and operated. Following the end-to-end spirit of the Internet SIP was supposed to turn down the walled gardens of PSTN networks and free communication services from the grip of large telecom operators. By moving the intelligence to the end systems, developers were supposed to be able to develop new communication services that will innovate the way we communicate with each other. This was to be achieved without having to wait for the approval of the various telecommunication standardization groups such as ETSI or the support of incumbent telecoms.

Session border controllers are usually implemented as SIP Back-to-Back User Agents (B2BUA) that are placed between a SIP user agent and a SIP proxy. The SBC then acts as the contact point for both the user agents and the proxy. Thereby the SBC actually breaks the end-to-end behavior of SIP, which has led various people to deem the SBC as an evil incarnation of the old telecom way of thinking. Regardless of this opposition, SBCs have become a central part of any SIP deployment.

In this paper we will first give a brief overview of how SIP works and continue with a description of what SBCs do and the different use cases for deploying SBCs.

2 A Short Introduction to SIP

By the mid nineties the Internet had established itself as a consumer product. The number of users buying PCs and subscribing to an ISP for a dial-up access was increasing exponentially. While mostly used for the exchange of Email, text chatting and distribution of information VoIP services based on proprietary solutions as well as H.323 started to gain some popularity.

While there is no organization that is formally responsible for the Internet as such the Internet Engineering Task Force (IETF) is playing the role of the standards organization of the Internet. The IETF has among others produced the specifications for the transport and routing of packets in the Internet as well as the protocols for Email, address resolutions and various other applications and services running on top of the Internet.

By the mid nineties the IETF had already produced different protocols needed for IP-based telephony services. The Real-Time Transport Protocol (RTP) [1] enabled the exchange of audio and video data. The Session Description Protocol (SDP) [2] enabled the negotiation and description of multimedia data to be used in communication session. With the Session Announcement Protocol (SAP) [3] it was even possible to distribute the necessary information to watch a certain publicly broadcasted audio and video session. Further, the first applications, mostly open source, for the sending and reception of real-time audio and video data were available.

In those days, the procedure for establishing a VoIP call between two users based on the IETF standards would look as follows: The caller starts his audio and video applications at a certain IP address and port. The caller then either calls the callee over the phone or sends him an Email to inform him about the IP and port address as well as the audio and video compression types. The callee then starts his own audio and video applications and informs the caller about his IP and port number. While this approach was acceptable for a couple of researches wanting to talk over a long distance or for demonstrating some research on QoS this was clearly not acceptable for the average Internet user.

The Session Initiation Protocol (SIP) [4] was the attempt of the IETF community to provide a signaling protocol that will not only enable phone calls but can be also used for initiating any kind of communication sessions. Hence, SIP can be used for VoIP just as well as for

setting up a gaming session or controlling a coffee machine.

The SIP specifications describe three types of components: user agents (UA), proxies and registrar servers. The UA can be the VoIP application used by the user, e.g., the VoIP phone or software application. A VoIP gateway, which enables VoIP users to communicate with users in the public switched network (PSTN) or an application server, e.g., multi-party conferencing server or a voicemail server are also implemented as user agents.

The registrar server maintains a location database that binds the users' VoIP addresses to their current IP addresses.

The proxy provides the routing logic of the VoIP service. When a proxy receives a SIP request from a user agent or another proxy it also conducts service specific logic, such as checking the user's profile and whether the user is allowed to use the requested services. The proxy then either forwards the request to another proxy or to another user agent or rejects the request by sending a negative response.

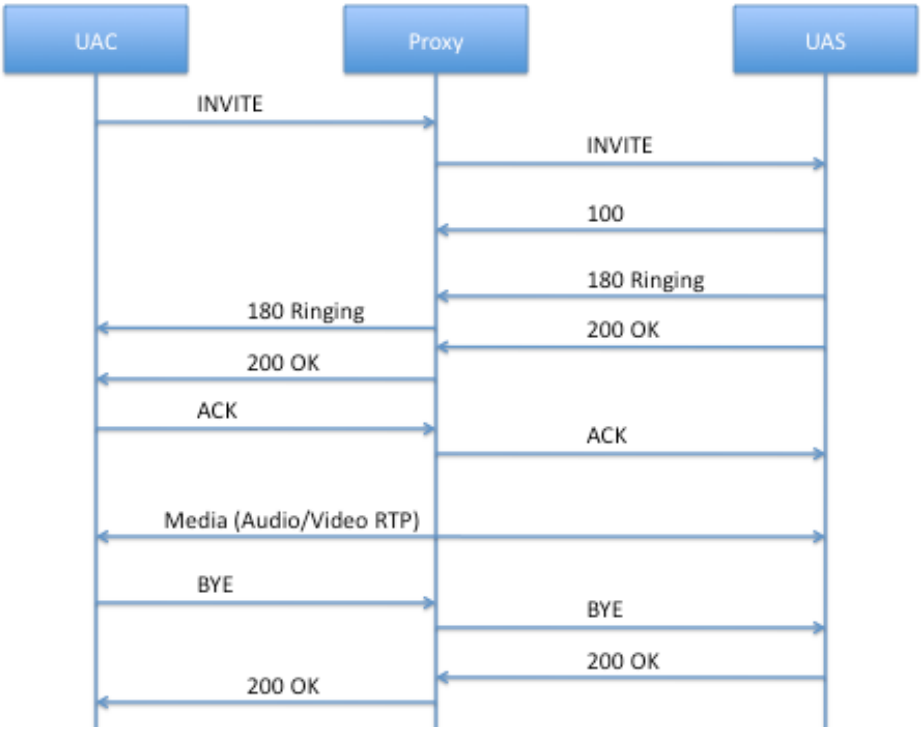

Figure 1 SIP Call flow

With regard to the SIP messages we distinguish between requests and responses. The INVITE request is used to initiate a dialog between two users. A BYE request is used for terminating this dialog. Responses can either be final or provisional. Final responses can indicate that a request was successfully received and processed by the destination.

Alternatively, a final response can indicate that the request could not be processed by the destination or by some proxy in between or that the session could not be established for some reason. Provisional responses indicate that the session establishment is in progress, e.g. the destination phone is ringing.

In this paper we distinguish three types of SIP message exchanges, namely registrations, dialogs and out of dialog transactions.

A SIP registration enables a user agent to register its current address, IP address for example, at the registrar. This enables the registrar to establish a correlation between the user agent's permanent address, e.g. sip:user@frafos.com, and the user agent's current address. In order to keep this correlation up to date the user agent will have to repeatedly refresh the registration. The registrar will then delete a registration that is not refreshed for a while.

A SIP dialog, a call for example, usually consists of a session initiation phase in which the caller generates an INVITE that is responded to with provisional and final responses. The session initiation phase is terminated with an ACK, see Figure 1. A dialog is terminated with a BYE transaction. Depending on the call scenario the caller and callee might exchange a number of in-dialog requests such as reINVITEs or REFER.

The last type of SIP interactions is SIP transactions that are not generated as part of a dialog. These out of dialog messages can be observed when the SUBSCRIBE and NOTIFY requests are exchanged between two SIP user agents [4]. This is the case when a SIP node wants to be informed about a certain event. In this case this node sends a SUBSCRIBE request to the server in charge of this event. Once this event occurs, the server will send a NOTIFY request to the SIP node carrying information about the event. Other out of dialog SIP requests include OPTIONS and INFO that are often used for exchanging information between SIP nodes or as an application level heartbeat.

Every SIP message consists of three parts: First line, message header and message body, see Figure 2. The first line states the purpose of the message. For requests it identifies its type and the destination address. For replies the first line states the result as a numerical 3-digit status code together with a textual human-readable form. The second part of the message, the header part, includes a variety of useful information such as identification of the User Agent Client and the SIP path taken by the request. The third part includes a message body

that contains application specific information. This can be for example session description information (SDP) indicating the supported codecs.

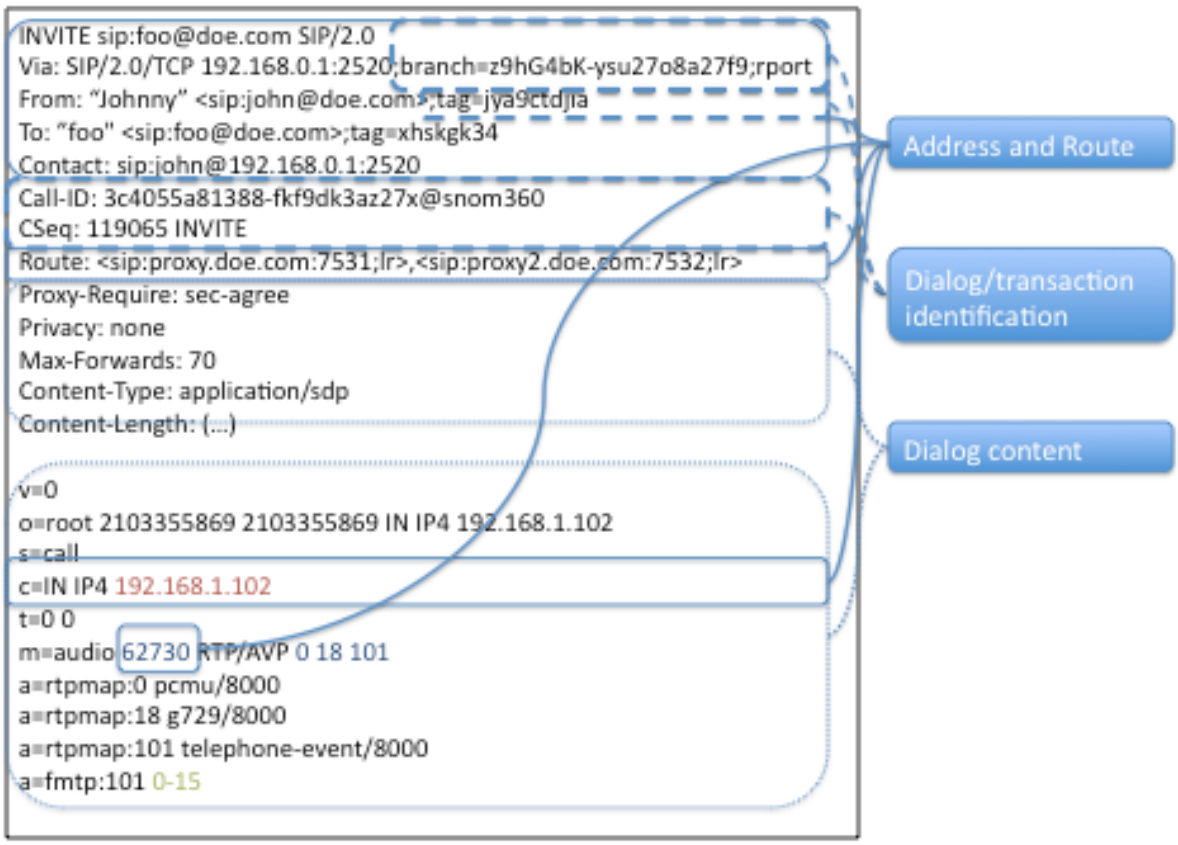

Figure 2 Content of SIP messages

The information contained in these three parts can be roughly divided into three categories, see Figure 2:

- Addressing and routing information: This includes information about who has sent the message, and where it is destined to, the next hop to be sent to as well as the hops it has traversed. This information is included in the first line as well as in different headers such as From, To, Contact, P-Asserted-Identity header, Via, Route, Path and others. The message body can contain information about where the media traffic should be sent to or is expected to come from.
- Dialog and transaction identification: This part of the SIP messages is used to uniquely identify a SIP dialog or transaction. This information is included in SIP headers such as Cseq, Call-Id as tags included in From, To and Via headers.
- Dialog content: With dialog content we categorize data that is included in a SIP

message that is either used to describe certain features of a dialog or indicate how a node receiving the message should process the message. This can include parts of the SIP message body carrying SDP, which includes description about which audio or video codes to use. Certain headers such as Privacy for example indicate the user's wishes with regard to the way private information such as user address should be handled.

3 What Do SBCs Do?

Since their introduction nearly 10 years ago, SBCs have been increasingly used to accomplish an increasing set of requirements [6]. This section will start with a brief why SBCs emerged and an overview of the general behavior of SBCs followed by a more detailed look on how an SBC provides different features such as NAT traversal or denial of service protection.

3.1 SIP Design Shortcomings and Emergence of SBCs

It is important to understand that despite all effort a protocol standard is hardly ever perfect. There are numerous reasons for that.

Standards attempt to combine conventional wisdom with innovation spirit. There are different contributors with different mindsets and objectives. And importantly people simply err. The result frequently leads to various workarounds and standard updates, a process which is largely similar to legislation. With SIP, various design shortcomings in fact induced a whole aftermarket: Session Border Controllers.

Probably the single biggest mistake in SIP design was ignoring the existence of NATs. This error came from a belief in IETF leadership that IP address space would be exhausted more rapidly and would necessitate global upgrade to IPv6 and eliminate need for NATs. The SIP standard has assumed that NATs do not exist, an assumption, which turned out to be a failure. SIP simply didn't work for the majority of Internet users who are behind NATs. At the same time it became apparent that the standardization life-cycle is slower than how the market ticks: SBCs were born, and began to fix what the standards failed to do: NAT traversal.

Yet other source of mistakes has been the lack of a clear data model behind the protocol design. Numerous abstract notions, such as dialog or session, transaction or contact simply didn't have unique unambiguous identifiers associated with them. They were calculated or almost guessed out of various combinations of header-fields, decreasing the interoperability. Some message elements, such as Call-ID, have been overloaded with multiple meanings. While some of these were fixed in the later SIP revision and its extensions (rport, branch, gruu, session-id) the market forces jumped in quickly. SBCs began to implement "protocol repair".

The other class of mistakes emerged from implementations. Many SIP components were built under a simplifying assumption that security comes for free. Numerous implementations were found to be vulnerable to malformed SIP messages or excessive load. The SBCs began to play a security role.

Over several years, Session Border Controllers became a de facto standard for which ironically no normative reference existed Session Border Controllers handle NATs, fix oddities in SIP interoperability and filter out illegitimate traffic. They began to incorporate elements of the standardized SIP components. For example, routing functionality contemplated by the standards for proxy servers, is nowadays part of reasonable SBC products. Similarly the SBCs often incorporate media recording and processing function, whether that's for quality assurance, archiving or legal-compliance purposes.

3.2 General Behaviour of SBCs

Figure 3 depicts the message flow of an INVITE request between a caller and a callee. This is the simplest message sequence that one would encounter with only one proxy between the user agents. The proxy's task is to identify the callee's location and forward the request to it. It also adds a Via header with its own address to indicate the path that the response should traverse. The proxy does not change any dialog identification information present in the message such as the tag in the From header, the Call-Id or the Cseq. Proxies also do not alter any information in the SIP message bodies. Note that during the session initiation phase the user agents exchange SIP messages with the SDP bodies that include addresses at which the agents expect the media traffic. After successfully finishing the session initiation phase the user agents can exchange the media traffic directly between each other without the involvement of the proxy.

SBCs come in all kinds of shapes and forms and are used by operators and enterprises to achieve different goals. Actually even the same SBC implementation might act differently depending on its configuration and the use case. Hence, it is not easily possible to describe an exact SBC behavior that would apply to all SBC implementations. However, in general one we can still identify certain features that are common for most of SBCs. For example, most SBCs are implemented as "Back-to-Back User Agent" (B2BUA).

A B2BUA is a proxy-like server that splits a SIP transaction in two pieces: on the side facing User Agent Client, it acts as server; on the side facing User Agent Server it acts as s client.

While a proxy usually keeps only state information related to active transactions, B2BUAs keep state information about active dialogs, e.g., calls. That is, once a proxy receives a SIP request it will save some state information. Once the transaction is over, e.g., after receiving a response, the state information will soon after be deleted. A B2BUA will maintain state information for active calls and only delete this information once the call is terminated.

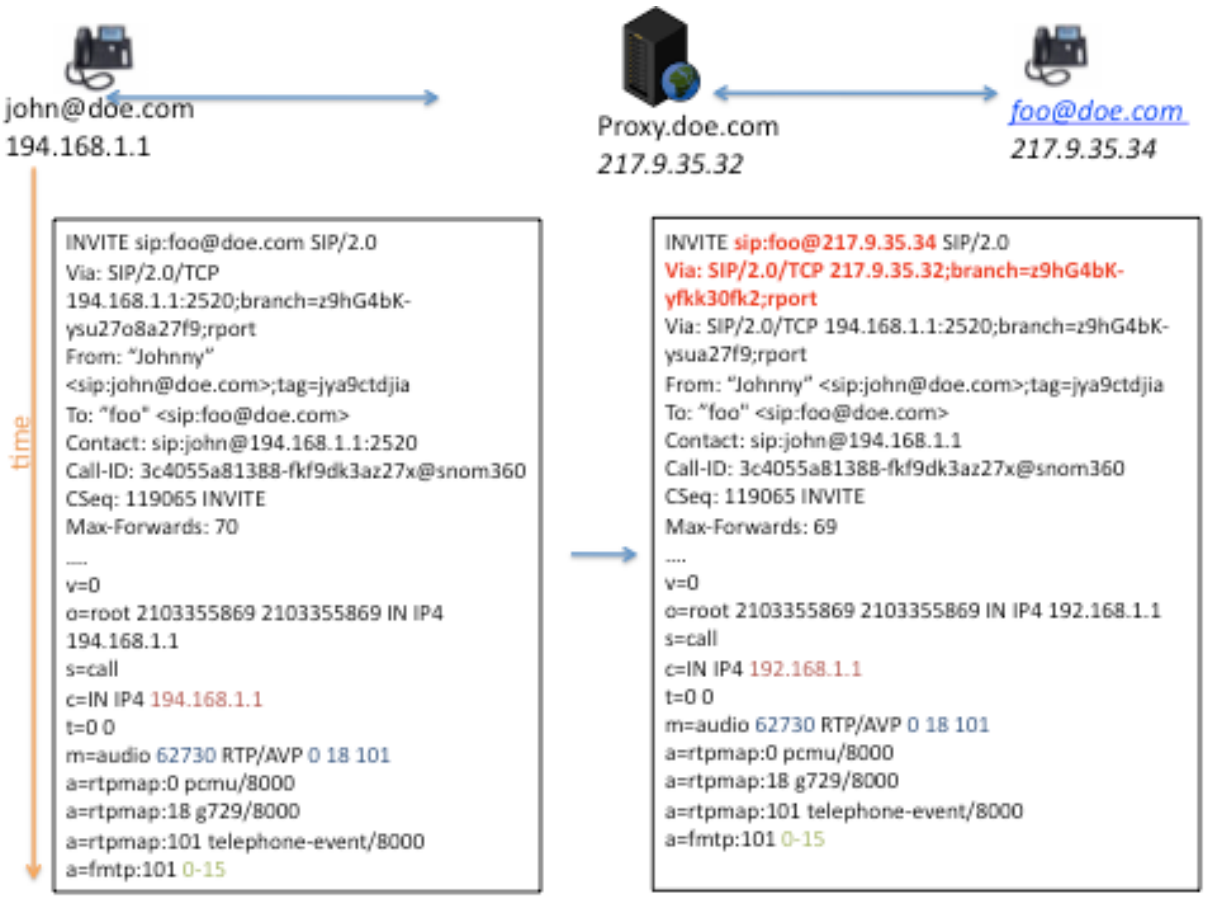

Figure 3 Purist SIP call flow

Figure 4 depicts the same call flow as in Figure 3 but with an SBC in between the caller and the proxy. The SBC acts as a B2BUA that behaves as a user agent server towards the caller and as user agent client towards the callee. In this sense, the SBC actually terminates that call that was generated by the caller and starts a new call towards the callee. The INVITE message sent by the SBC contains no longer a clear reference to the caller. The INVITE sent by the SBC to the proxy includes Via and Contact headers that point to the SBC itself and not the caller. SBCs often also manipulate the dialog identification information listed in the Call-Id and From tag. Further, in case the SBC is configured to also control the media traffic

then the SBC also changes the media addressing information included in the c and m lines of the SDP body. Thereby, not only all SIP messages will traverse the SBC but also all audio and video packets. As the INVITE sent by the SBC establishes a new dialog, the SBC also manipulates the message sequence number (CSeq) as well the Max-Forwards value.

Note that the list of header manipulations listed in *Figure 4* is only a subset of the possible changes that an SBC might introduce to a SIP message. Furthermore, some SBCs might not do all of the listed manipulations. If the SBC is not expected to control the media traffic then there might be no need to change anything in the SDP header. Some SBCs do not change the dialog identification information and others might even not change the addressing information.

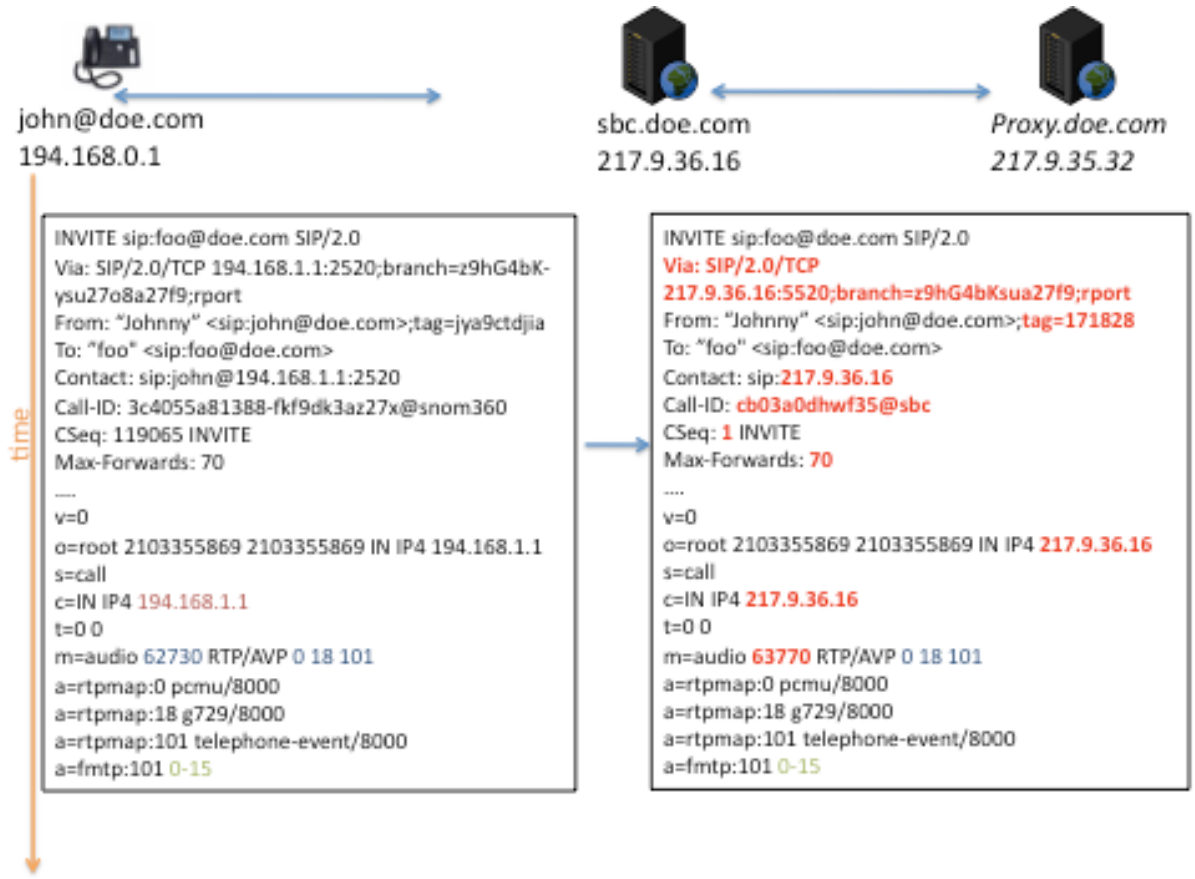

Figure 4 SIP call flow with SBC

3.3 Topology Hiding

As the result of a SIP session establishment the involved end points will know the IP addresses of where to send and receive media traffic. This means that a user using SIP for

calling a PSTN number will know the IP address of the PSTN gateway that is responsible for bridging the VoIP service with the PSTN. Further, during the session establishment phase all the involved proxies will include their addresses in the Via headers.

A malicious user could use this information to either attack an operator's proxies or even get access to the PSTN gateways directly. By having the ability to contact the PSTN gateways directly, an attacker might be able to misuse any security holes that might exist at the PSTN gateway. This would allow the attacker to initiate calls to the PSTN with the costs being incurred on the operator.

To hide the internal components of an operator, all messages leaving the operator's network would traverse an SBC. The SBC replaces the addresses of internal components with its own. Hence, headers such as Contact, Via, Record-Route, Route and so on would include the SBCs address only.

To hide the address of PSTN gateway or application servers, the SBC would include its own address in the SDP part of the SIP messages.

3.4 NAT-Traversal Support

Network Address Translators (NAT) are used to overcome the lack of IPv4 address availability by hiding an enterprise or even an operator's network behind one or few IP addresses. The devices behind the NAT use private IP addresses that are not routable in the public Internet.

In case a user agent is located behind a NAT then it will use a private IP address as its contact address in the Contact and Via headers as well as the SDP part. This information would then be useless for anyone trying to contact this user agent from the public Internet.

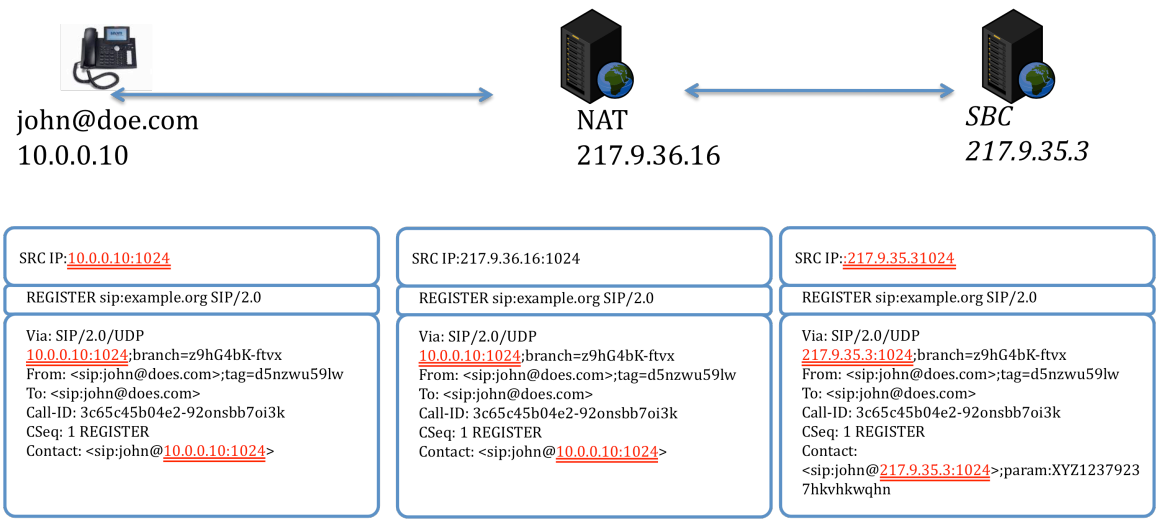

Figure 5 SBC and NAT traversal: Registration handling

There are different NAT traversal solutions such as STUN [7] and ICE [8]. Which solution to use depends on the behavior of the NAT and the call scenario. When using an SBC to solve the NAT traversal issues the most common approach for SBC is to act as the public interface of the user agents. This is achieved by replacing the user agent's contact information with those of the SBC.

In order for a user agent to be reachable through the public interfaces of an SBC, the SBC will manipulate the registration information of the user agent. The user includes its private IP address as its contact information in the REGISTER requests. Calls to this address will fail, since it is not publicly routable. The SBC replaces the information in the Contact header with its own IP address, see Figure 5. This is the information that is then registered at the registrar. Calls destined to the user will then be directed to the SBC. In order for the SBC to know which user agent is actually being contacted the SBC can keep a local copy of the user agent's registration. The local copy includes the private IP address and the user's SIP URI as well as the public IP address included in the IP header that was assigned to the SIP message by the NAT.

Alternatively the SBC can store this information in the forwarded SIP messages. This is displayed in Figure 5 with the user's contact information combined in a special format and added as an additional parameter to the Contact header. The contact information would include the user's private IP address and SIP URI as well as the public IP address in the IP header of the SIP message. When the registrar receives a request for the user, the registrar will return the complete contact information to the proxy, which will include this

information in the SIP message. The SBC can then retrieve this information from the SIP request and use it to properly route the request to the user.

Adding the user agent's contact information to the registered contact information has many advantages. As the SBC does not have to keep local registration information this solution is simple to implement and does not require memory for keeping the information. Further, requests destined to the user agent do not necessarily have to traverse the SBC that has processed the user agent's registration messages. Any SBC that can reach the user agent can correctly route messages destined to the user agent based on the information included in the SIP request. This advantage applies, however, only in some cases. In case the NAT used in front of the user agent accepts traffic only from the IP addresses which the user agent has contacted previously then only the SBC that has processed the user agent's REGISTER requests will be able to contact the user agent.

Keeping a local copy of the registration information increases the processing requirements on the SBC. The SBC will have to manage a local registration database. Beside the memory requirements the SBC will have to replicate this information to a backup system if it is to be highly available. This will further increase the processing requirements on the SBC and increase the bandwidth consumption.

However, keeping a local copy of the registration information has its advantages as well. When receiving a message from a user agent a network address translator binds the private IP address of the user agent to a public IP address. This binding will remain active for a period of time –binding period. In case the user agent does not send or receive any messages for a period of time longer than the binding period then the NAT will delete the binding and the user agent will no longer be reachable from the outside. To keep the binding active, the user agent will have to regularly refresh it. This is achieved by sending REGISTER requests at time intervals shorter than the binding period. As REGISTER messages have to be usually authenticated, having to deal with REGISTER messages sent at a high frequency would impose a high performance hit on the operator's infrastructure. SBCs can help to offload this load. When a user agent sends the first REGISTER request, the SBC forwards the REGISTER request to the operator's registration servers. Once the registration was successfully authenticated and accepted by the operator, the SBC will keep a local copy of the registration information. Instead of forwarding each incoming REGIETER request to

the operator's registration servers, the SBC will only send REGISTER requests to the registration servers at rather large time intervals (in the range of hours). Registration requests arriving from the user agent that do not change the content registration information will be replied to by the SBC itself. The SBC will also inform the registration server once the local registration expires or changes.

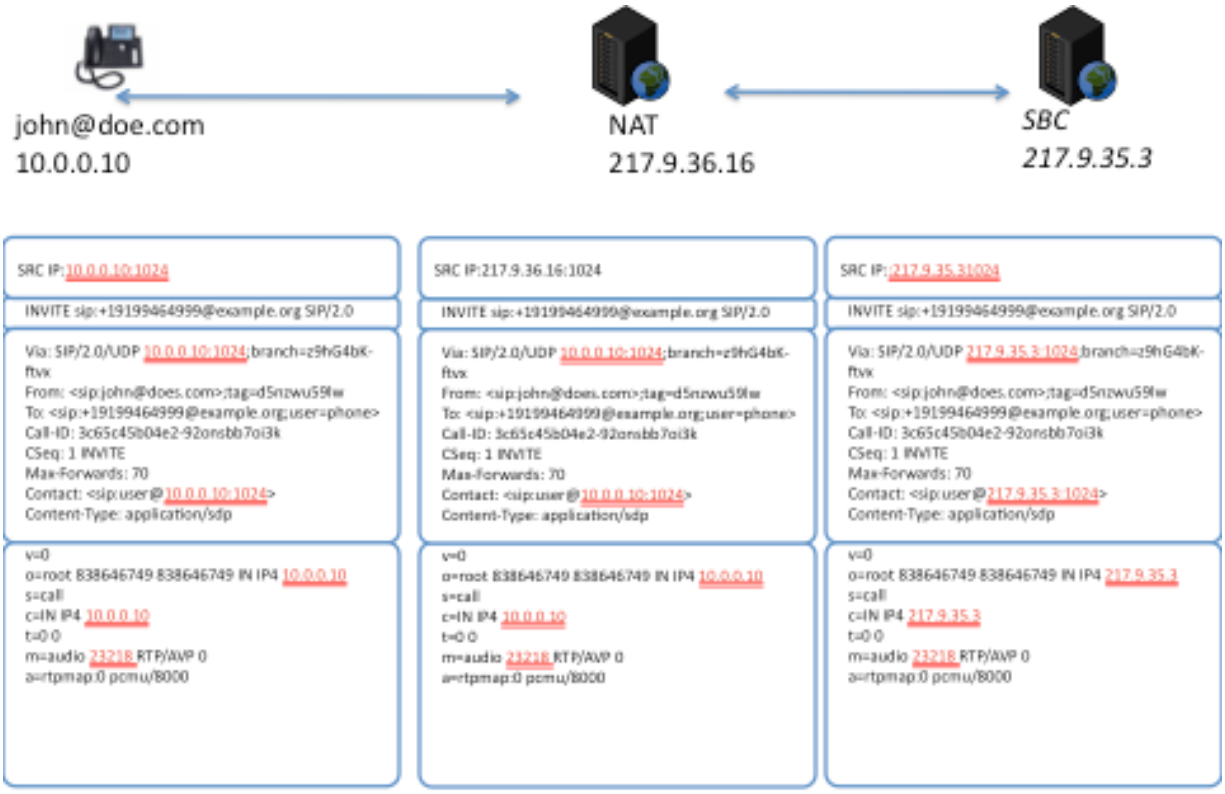

Figure 6 SBC and NAT traversal

Similar to the registration case, the SBC will also include itself in the path of INVITE and other request messages, see Figure 6. When receiving an INVITE from a user agent behind a NAT, the SBC will include a Via header with its own address, replace the information in the contact header with its own address and also replace the address information in the SDP body with its own address. Thereby, all SIP messages and media packets will traverse the SBC.

3.5 NAT Traversal and Media

While NAT traversal of SIP messages may appear complicated after all, the yet more complex task is enabling media to traverse NATs. The initial problem statement is the same.

If SIP devices behind NATs advertise their IP addresses, their peers on the other side of NATs cannot route traffic to them.

The solution SBCs came with simply ignores the way SIP works. Instead of sending media to the IP address and port number advertised in the SIP SDP bodies, SBCs send media for a user agent symmetrically back to where the agent has sent its own media from. This symmetric communication typically works because it is the traffic pattern NAT manufactures have been used to before the arrival of VoIP.

It is important to know that while this mostly works, it has several limitations. First of all, it only works with clients that are built "symmetric way", i.e., they use the same port for sending and receiving media. Nowadays that's fortunately the majority of available equipment.

The other noticeable disadvantage is "triangular routing": an SBC must relay all VoIP traffic for a call, to make the paths caller-SBC and SBC-callee symmetric. That is in fact quite an overhead for a VoIP operator. With the most common codec, G.711, a relayed call consumes four 87.2 kbps streams: two outbound, two inbound.

Some other disturbing limitations may occur too. For example, if a SIP device uses Voice Activity Detection (VAD) and fails to send any voice packets initially, the SBC will not learn its address and will not forward incoming media to it as well. Also some NATs are simply built in such a poor way, that the only thing which almost always works is HTTP and SIP just fails.

Despite these limitations, SBCs have solved the "NAT problem" in a vast majority of use-cases.

3.6 Denial of Service and Overload Protection

Like any other Internet-based service VoIP servers can be the target of denial of service attacks.

Attacks can be disguised as legitimate VoIP traffic so distinguishing between a denial of service attack or a sudden surge in traffic due to some event is not always possible. Hence, VoIP operators need to incorporate mechanisms that monitor the load and the incoming traffic, identify the overloaded resources and the cause of the overload and react in a manner that will prevent a complete service interruption.

In order to keep the malicious traffic and overload away from the core servers, e.g. applications servers, proxies and PSTN gateways, there might be protection mechanisms located at the SBCs. In this context one can often find SBCs offering some or all of the following features:

- Traffic limitation: Operators can limit the rate of incoming calls and registrations. Once these limits are exceeded, the SBC starts rejecting messages arriving in excess of these limits. These limits can apply to single sources, e.g., accept no more than X REGISTER requests from source Y, to a range of senders or to all incoming traffic.
- Dynamic blacklisting: Static blacklists are usually used to drop traffic from certain sources without having to process it first. However, not all possible malicious sources are known in advance. Therefore, SBCs often monitor the incoming traffic and if certain characteristics were identified then user agents are dynamically added to a blacklist. These characteristics can be the number of messages sent by a source over a period of time, the content of the messages or the distribution of the called destinations –e.g., a source that calls a lot of different destination in a row is very likely to be scanning the network in search for a destination with some weakness. Once a source is blacklisted all messages from that source would be rejected or dropped.
- Content filtering: An attacker could try to get access to some protected resources by launching an SQL injection attack or try to bring a server down by sending SIP messages with malformatted content. By analyzing the content of incoming SIP messages and rejecting messages that seem to include malicious content, the SBCs can protect the core components of the network.
- Caller prioritization: Customers of a VoIP service expect that their provider will still handle their calls even under overload or attack scenarios. To achieve this an SBC can identify calls generated by registered customers of the operator by keeping a local registration database. Under overload scenarios the SBC would then only accept calls originating from registered users.

3.7 Regulatory Features

With the increased success of VoIP services, providers of VoIP services will have to consider an issue that the Internet has managed to successfully ignore for a long time, namely legal

regulations. The traditional telecom market is one of the most regulated market segments. Current regulations describe in great detail how an emergency call must be dealt with in the network and how to intercept the call of a wrong doer.

To be able to support lawful interception an operator requires access to both signaling and media traffic. VoIP providers that do not offer IP access and use the SIP call establishment model described in Figure 1 have only access to the signaling information. By using an SBC for controlling both signaling and media packets the operator has an obvious node for supporting lawful interception.

3.8 Access Control and Fraud Prevention

As the name already implies, SBCs are tasked with controlling which users and what messages can cross the borders of a VoIP infrastructure and use the offered VoIP services. Most SBCs will offer most if not all of the following mechanisms:

- White/Blacklists: By maintaining lists of trusted and untrusted users and sources an SBC can easily determine whether a certain message should be accepted or rejected without further processing.
- Media control: As described in Sec. 3.1 SBCs often replace the addresses included in the SDP parts with their own. On the one hand this is needed for supporting NAT traversal. On the other hand this enables the SBC to ensure that only users that have successfully established a call –e.g., their INVITE requests were accepted by the callee- are allowed to send media traffic. This way an SBC can prevent a malicious user from contacting a PSTN gateway or an application server directly.
- Fraud prevention: Prices for a flat rate service are determined based on a certain expected user behavior. However, operators often face the case that a user subscribes for a flat rate telephony residential service but then starts reselling telephony minutes. This kind of behavior causes financial losses to the operator and overloads the network. To suppress this fraud possibility, operators can use SBCs to limit the number of parallel calls generated by a user as well as the duration and frequency of calls. Anything beyond observing the parallel number of calls made by a user would, however, require the SBC to keep track of the user's behavior over longer time periods. This might be better delegated to specialized fraud detection solutions. The SBC would then feed the fraud detection solution with call detail

records. The fraud detection solution analyzes these records and based on the user subscription profile as well as various policies decides on whether a certain user is trying to cheat the system. If that is the case then the fraud detection solution informs the SBC about which user to block. This naturally requires the SBC to offer open interfaces for the communication with fraud detection solutions.

3.9 Interoperability Mediation

There are different standardization groups working on SIP. Different developers often interpret the same specifications differently. This means that interoperability between SIP products of different vendors is unfortunately not always guaranteed.

SBCs often have the capability to overcome some of these interoperability problems by manipulating the content of SIP messages so that they better fit the expectations of the receiving side. One can distinguish between three interoperability issues; namely SIP flavors, SIP content and transport protocols.

3.9.1 SIP Flavours

SIP is being used in both mobile and fixed networks as well as a transition protocol in the 3GPP R4 [10] release. In the ISP environment, SIP as was specified by the IETF [4] is used mostly. In the fixed environment, the TISPAN specifications [11] are used. In the mobile network environment the 3GPP IMS specifications [12] are the most favored. SIP-I [10] is proposed for trunking scenarios in which SIP is used as the signaling protocol used to connect SS7 based networks over an IP core network.

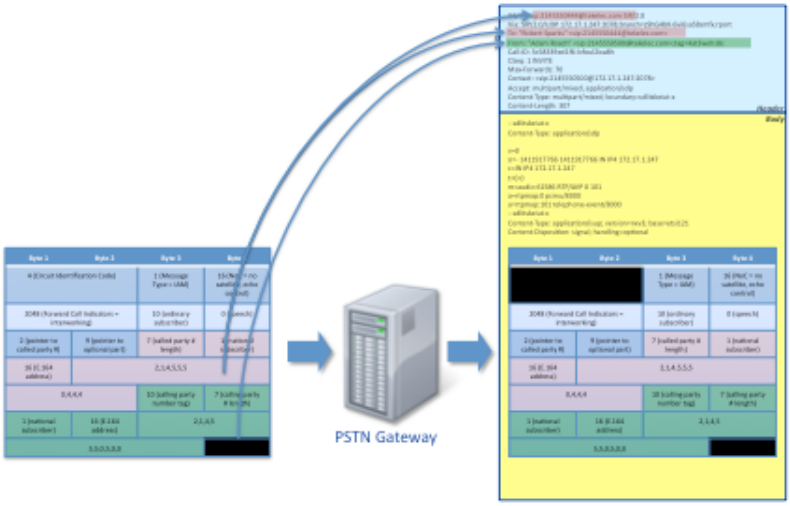

Figure 7 SIP-I generation at an MSS (PSTN gateway)

The differences between the SIP specifications from IMS, IETF and TISPAN are mainly restricted to the addition of certain headers, authentication mechanisms and usage of certain SIP extensions such as NOTIFY/SUBSCTIBE or certain XML bodies.

Besides the differences in the SIP headers, SIP-I adds another body type to the SIP message; namely an ISUP part, which is added by a PSTN gateway after generating a SIP message from an incoming SS7 message, see Figure 7. This ISUP body is then used by the receiving PSTN gateway for reconstructing the SS7 signaling messages towards the other part of the call.

In the context of interoperability of SIP flavors, SBCs can provide the following services:

- Stateless SIP header manipulation: An SBC can be configured to remove certain headers and add others. This way, an SBC can for example delete headers that are useful in an IMS or TISPAN but not in an IETF SIP environment.
- Statefull message handling: Different SIP based deployments might expect different call flows. So while an ISP using SIP according to the IETF RFC3261 specification a mobile operator might be deploying the IMS specifications [12]. One of the major differences between the two specifications is that IMS deployments heavily rely on provisional acknowledgments. (PRACK) –that is a user agent server sending a provisional response expects an acknowledgement from the user agent client that

the response was correctly received [13]. As the capability of generating PRACK requests is not widely used in IETF based deployments an SBC on the border between the ISP and the mobile operator could mediate between the two call flows by generating the appropriate PRACK requests, see Figure 8.

- Message blocking: Certain SIP messages might be useful in one network as they provide a certain service. However, if this service is not provided across the interconnection points then exchanging them across the networks does not make sense. SBCs can be configured to reject certain messages such as NOTIFY if presence services are not provided across the network for example.
- SIP-I to SIP manipulation: SIP-I requests carry an ISUP part as part of the SIP body. This could cause problems for SIP components that do not understand ISUP and do not expect to see such information in a SIP message. Some SBCs can overcome this issue by removing the ISUP part when forwarding a message to the SIP side of the communication and adding the appropriate ISUP body before forwarding a message to the SIP-I part of the call. This will often require some understating of ISUP and keeping ISUP related state information.

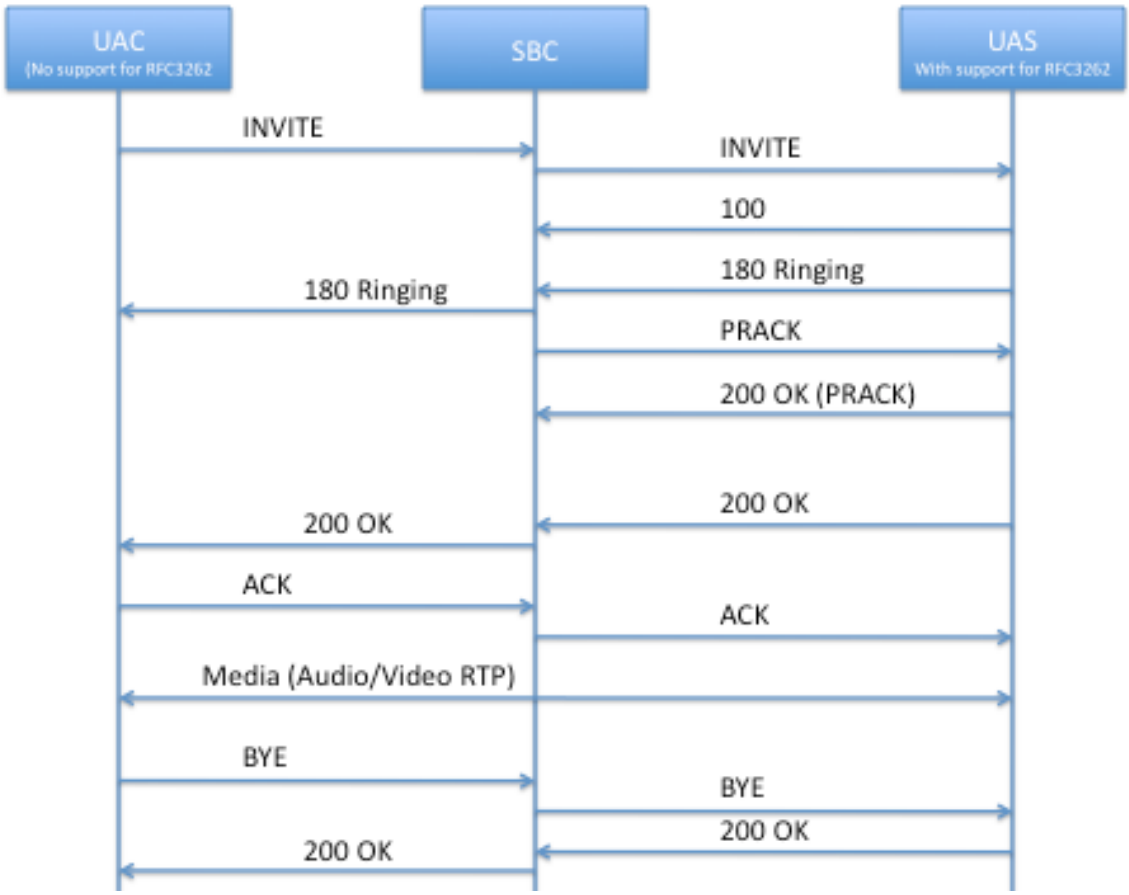

Figure 8 Mediation between IMS and non IMS UAs

3.9.2 SIP Content

The experience from various interoperability events shows that different vendors interpret the SIP specifications slightly differently. Especially parts that are specified with the strength of "SHOULD" or "MAY" are often implemented as a "MUST" or ignored completely. This makes the communication between two components from different vendors sometimes impossible.

Some SBCs can be configured to overcome some of these issues and to fix certain issues that cause these interoperability problems by offering some of the following features:

- Existence of certain headers: Some SIP components expect to see certain SIP headers with certain information, for example a Route header pointing to them. Others might not bother to add this header. An SBC offering mediation service can be configured to take these special interpretations of the implementers into account before forwarding a request and add or remove problematic headers.

- Location of information: Some SIP components expect to see their address in the Request-URI whereas others want to see it in the Route header or both. This might not always be how the location information is included in the SIP request especially if a request was redirected from one component to another.
- Tags and additional information: Again some SIP components might expect to see certain tags attached to certain headers such as rport with a Via header whereas other SIP components might not add them. By introducing an SBC with support for mediation in between the incompatible components these problems can be fixed.

3.9.3 SIP Transport

SIP can be transported over UDP, TCP and SCTP. Further, it can work over IPv4 and IPv6. The capabilities of different SIP implementations might vary with this regard. That is, some components could support UDP but not TCP and others prefer to use SCTP. An SBC could solve interworking problems here in the following manner:

- Use the transport protocol preferred by a destination for sending requests to it.
- Use the IP version preferred by a destination for sending requests to it. That is, use IPv4 on one part of the call and IPv6 on the other part.
- Translate between the incoming and outgoing transport protocol

3.9.4 Media Transcoding

Especially on the borders between fixed and mobile networks there might be some need for transcoding the audio or video compression system from one format to the other. Different SBCs offer the possibility to integrate specialized transcoding hardware. For the case when the expected need for transcoding is low, some SBCs offer software based transcoding solutions.

4 SBC Deployment Scenarios

While some SBCs might actually offer all of the features listed in Sec. 3 in most cases different SBCs will offer a different set of features depending on their intended use cases. In general, one can identify three use cases for SBCs; namely on the border between an operator and its subscribers, on the border between two operators and on the borders of an enterprise, see Figure 9.

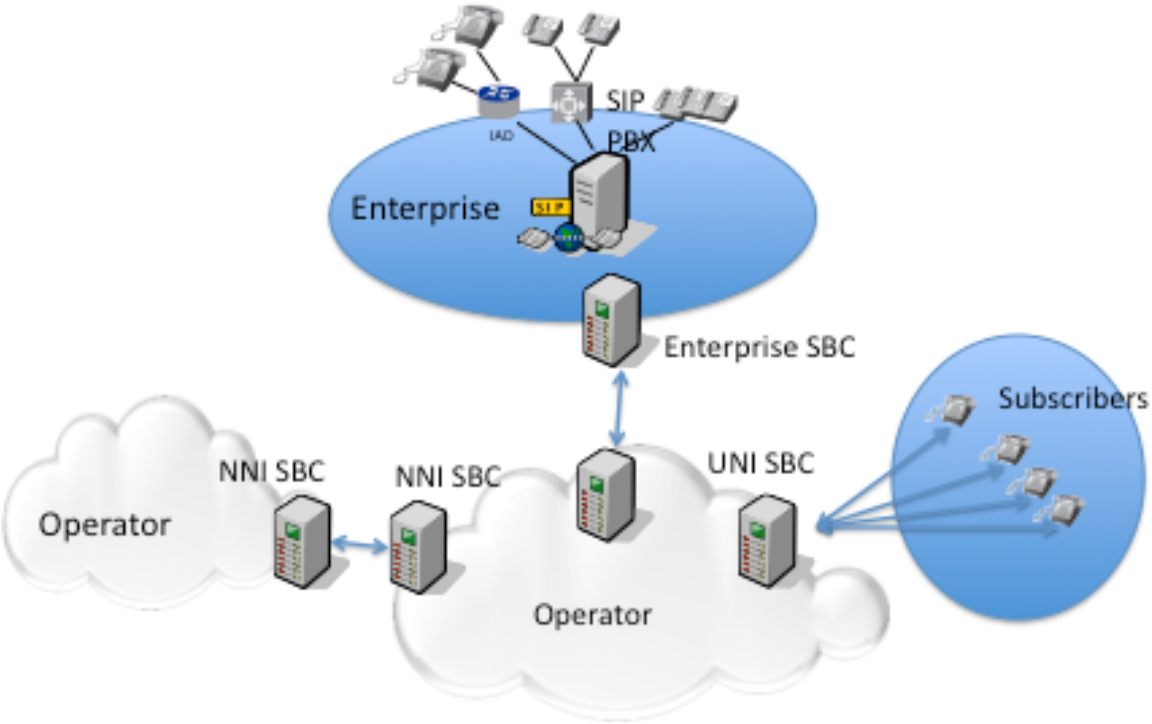

Figure 9 SBC Deployment scenarios

4.1 User-Network-Interface (UNI) SBC

Operators use SBCs to establish a secure border between their core VoIP components–e.g., PSTN gateway, SIP proxy and application servers- and their subscribers. Actually this usage scenario is the oldest use case of SBCs.

In terms of the SBC features discussed in Sec. 3 one can expect the following:

- Topology hiding: One of the most widely observed fraud scenarios is the case of a malicious user detecting the address of a PSTN gateway and accessing that gateway directly. Once the attacker has managed to access the gateway he can start selling telephony minutes through that gateway. To avoid this kind of attacks UNI SBCs are expected to hide the details of the operator's network.
- NAT traversal: Subscribers are often located behind a NAT. Hence, a UNI SBC is expected to support solutions for NAT traversal.
- Denial of service and overload protection: If we have learned one thing from the Internet, then it is that there will always be some people with enough technical skills and time to figure out a way to attack some service. In order to protect VoIP services from a DoS attack or a sudden increase in the number of calls, e.g., Christmas calls, a UNI SBC will have to offer DoS detection and prevention mechanisms. This can include dynamic blacklisting, prioritization of registered users and traffic limiting.
- Regulatory features: The SBC will very likely be the only point in the operator's network, which will route both the signaling and media packets of the user. Hence, a UNI SBC is the ideal place for supporting regulatory features such as lawful interception.
- Access control and fraud prevention: An operator will most likely want to handle only calls generated by or destined to its own subscribers. A UNI SBC has thus the task of identifying relevant calls and rejecting non-relevant calls. Further, in order to detect fraud and misuse, UNI SBCs will often collect per user data such as the number of calls generated by a subscriber. Based on the collected data the SBC would then execute certain policies like dropping calls generated in excess of certain limits.
- Interoperability mediation: UNI SBCs will have to communicate with a large number of subscribers using VoIP user agents from different vendors. However, different vendors might implement things differently or introduce certain implementation bugs. A UNI SBC should shield the operator's core components from this diversity. Aspects of SIP to SIP-I or IMS support would however play less of a role for a UNI SBC.

- Capacity: In terms of capacity a UNI SBC can range from a small solution supporting a couple of hundred parallel calls to larger ones supporting thousands of calls. The required capacity depends on the size of the operator and number of subscribers. A UNI SBC is also expected to have to deal with a large number of TCP or TLS connections.

4.2 Network-Network-Interface (NNI) SBC

While an increasing number of operators have already replaced their SS7 based telecommunication core network with a SIP based solution, the interconnection to neighboring partners is still often realized over an SS7 peering point. This means that a call that is carried over a VoIP network is translated to an SS7 call and then possibly back to VoIP again. The translation requires specialized components and resources, which increases the network operation costs and introduces unnecessary processing delay. To avoid these costs and delays operators have started introducing SIP based interconnection points.

In order to establish a secure border to their neighbors operators will in general deploy an NNI SBC. NNI SBCs act mainly in the same way as described in Sec. 3, however, they can be adjusted for the following use cases:

- Topology hiding: Operators consider information about the topology of their networks as private information. Hence, an NNI SBC is expected to support topology hiding.
- NAT traversal: As NNI SBCs do not deal with subscribers directly NNI SBCs usually do not have to support features like NAT traversal or registration handling.
- Denial of service and overload protection: NNI SBCs establish peering relations to trusted neighbors. Hence, in terms of security, features related to traffic limiting and secure communication of TLS and IPSEC are of higher importance than fraud detection or dynamic blacklisting.
- Regulatory features: The SBC will very likely be the only point in the operator's network, which will route both the signaling and media packets of the user. Hence, a NNI SBC is the ideal place for supporting regulatory features such as lawful interception.
- Access control and fraud prevention: An operator will most likely want to handle

calls only from trusted peering partners. Hence, access control is mainly limited to black and white lists. Further, whether media traffic should be controlled will also depend on the level of trust between the peering partners and whether there is a need to monitor the exchanged media traffic. In terms of fraud prevention, the peering partners will monitor the sum of exchanged traffic between two peering partners. Rejecting and shaping of traffic will only be needed if some service level agreement was violated and would be applied to the aggregated traffic and not to the traffic of single users.

- Interoperability mediation: NNI SBCs will communicate with a small number of peering partners. In general one could expect to clarify any interoperability issues on the SIP content level before deploying the peering relation. As this is not always possible having some level of support for the mediation of SIP content in NNI, SBC is desirable. Unlike the UNI case, an NNI SBC will have to deal with the aspects of SIP to SIP-I or IMS support.
- Capacity: In terms of capacity an NNI SBC will have to deal with a large number of parallel calls that will be, however, most likely received over a small number of TCP and TLS connections. Hence, NNI SBCs have to offer a high performance and be scalable to rapidly increase the available capacity without having to introduce a lot of changes in the peering structure. That is, the addition of another NNI SBC server should not lead to service interruption or require the coordination with the peering partners. This might require using a load balancer in front of the NNI SBCs that hides a cluster of NNI SBC servers behind a single IP address. This would then enable an operator to gradually increasing the number of NNI servers without having to communicate to its peering partners the IP addresses of the added SBCs.

4.3 Enterprise SBCs

Enterprises are increasingly replacing their PBXs with VoIP PBX or are extending their PBX with a VoIP module to benefit from attractive VoIP minute prices. Enterprise SBCs are used to secure the access to the PBX. The enterprise SBC is also expected to secure the communication to the VoIP operator, which is offering the VoIP service to the enterprise.

A VoIP PBX is a special case of a SIP user agent. On the one side the features and supplementary services supported by a PBX exceed by far the features supported by a VoIP

phone used by a residential subscriber.

In enterprise environments complex call flows such as call forwarding, music on hold or call parking are used much more often then in residential scenarios. This means that an enterprise SBC will have to deal with more complex call flows than a UNI or NNI SBC.

Further, one PBX will be providing VoIP service for more than one subscriber. A residential subscriber sends a single REGISTER request to inform the registrar server at his VoIP operator about its contact information. Using the same approach in an enterprise scenario would mean that a PBX has to possibly send hundreds if not thousands of registration messages depending on the number of served users in the enterprise. To avoid this avalanche of registrations the specification listed in [14] recommends either using static registrations or registration for multiple phones [14]. With the static registration, the registrar server of the VoIP operator would be pre-configured with the address of the PBX and the SIP addresses served by this PBX.

An SBC on the border of the enterprise network will have to support not only the general SIP specifications described in [4] but also the recommendations for connecting enterprises to an operator, see [14]. Further, an enterprise SBC would be expected to support the following:

- Topology hiding: The primary purpose of an enterprise SBC would be to hide the contact information of the PBX from the rest of the world.
- NAT traversal: In case there is a NAT between the PBX and the SBC then the enterprise SBC can offer NAT traversal support. In general one could, however, expect the enterprise SBC to act as a NAT. This is achieved by having two interfaces at the SBC with one of them private and the other public with the PBX connected to the SBC over the private one. The SBC would use its public address to communicate with the operator. Unlike NATs which only mangle the IP addresses in the IP headers, the SBC will use its public IP address in the contact, routing and body parts of the SIP message.
- Denial of service and overload protection: Enterprises can also become the targets of DoS attacks. Hence, appropriate mechanisms for detecting malicious behavior are needed. Further, [14] recommends using TLS or IPSEC for securing the

communication with the operator.

- Regulatory features: Enterprises do not offer public services and hence there is no need to support lawful interception.
- Access control and fraud prevention: In order to protect the PBX, an enterprise SBC should only accept calls arriving over a secure connection established with a trusted VoIP operator.
- Interoperability mediation: An enterprise SBC will communicate with a small number of components. Namely the enterprise PBX and the border elements of one or more VoIP operators. In general one could expect to clarify any interoperability issues on the SIP content level before connecting to a VoIP operator. As this is not always possible having some level of support for the mediation of SIP content in an enterprise SBC is desirable.
- Capacity: The expected capacity of an enterprise SBC will depend on the size of the enterprise. This might range from a single digit number of users up to thousands.

5 So SBCs Aren't Evil After All?

Session border controllers have become an integral part of VoIP solutions. Operators and enterprises deploy SBCs to secure their VoIP infrastructure and control the access to VoIP services.

So does this mean that the SBCs don't have any disadvantages?

SBCs break the end-to-end nature of SIP. As the SBCs work as B2BUAs they need to understand the received messages. This means that updates to SIP in terms of new SIP calls flows, headers and requests can often not be deployed until the deployed SBCs are upgraded to understand these updates. This can be a big obstacle in terms of innovation and establishes a new walled garden around VoIP deployments.

Further, breaking the end-to-end behavior of SIP calls makes debugging and monitoring of SIP calls much more difficult. As SBCs change the session identities of SIP messages, monitoring a SIP call requires collecting the SIP messages on both legs from the UAS to the SBC and the SBC to the UAC and correlating this information. This increases the complexity and costs of monitoring and quality assurance solutions.

SBCs not only break the end-to-end nature of the signaling path but also that of the media. When SBCs are used media packets are often no longer exchanged directly between the user agents but are routed through the SBC. This can have negative effects on the voice and video quality due to higher delays. Think about the case of two subscribers of a German VoIP provider on a visit to Australia. Calls between the two subscribers will be routed through Germany adding substantial delay to the audio packets. Also the operator will have to deal not only with signaling but also with media traffic, which consumes much more bandwidth. The faster Internet connection needed to carry the media traffic incurs more costs on the VoIP provider.

Ironically while one of the biggest drivers of using SBCs is the security aspect, security is also one of the major issues with SBCs.

Users wishing to encrypt or sign the content of their SIP message can use S/MIME [15]. S/MIME provides the necessary mechanisms for ensuring the integrity and confidentiality

of application level information such as Email or SIP messages.

When S/MIME is used for signing a SIP message, the sender generates a signature using his private key and adds the signature to the SIP message. The receiver of the message can check the authenticity of the message by decrypting the signature using the sender's public key.

To encrypt a message the sender uses the public key of the receiver. The encrypted message is then added as the body of a SIP message that contains only the minimal information needed for routing a SIP request. That is, all headers that are not needed for routing purposes as well as the SDP body would be encrypted. Only the receiver can decrypt the message using his own private key.

To do their job properly SBCs need to understand and read the SIP message headers and body. With an encrypted message this is only possible if the SBC decrypted the message first. This would, however, require the SBC to have access to the private key of the callee, which is in general not desirable. If access to the private key of the callee was possible then the SBC could then decrypt and then if needed encrypt the messages again before sending them to their final destination. Access to the private key of a callee is probably only possible in enterprise scenarios. In most other cases, having an SBC in the signaling path will result in call establishment failures.

As messages leaving an SBC differ from the ones received by the SBC, any signature included in a SIP request will no longer be valid after traversing an SBC. An SBC could naturally remove an existing signature and sign the request before forwarding it to the callee. This, however, would break the end-to-end security relation between the caller and callee and would replace it by a hop-by-hop one. The callee is expected to trust an SBC with whom he might not have a trust relation.

A similar issue can be observed when using the approach described in [17] for providing authenticated identities. In [17] a new entity called the authentication service is defined. The authentication service can be expected to be part of the SIP proxy but can be part of any component that has the capability of authenticating the users and posses the private key of the service provider. If an end device possesses the private key of the service provider, which might be the case for application servers or PSTN gateways, then the end device can act as the authentication service.

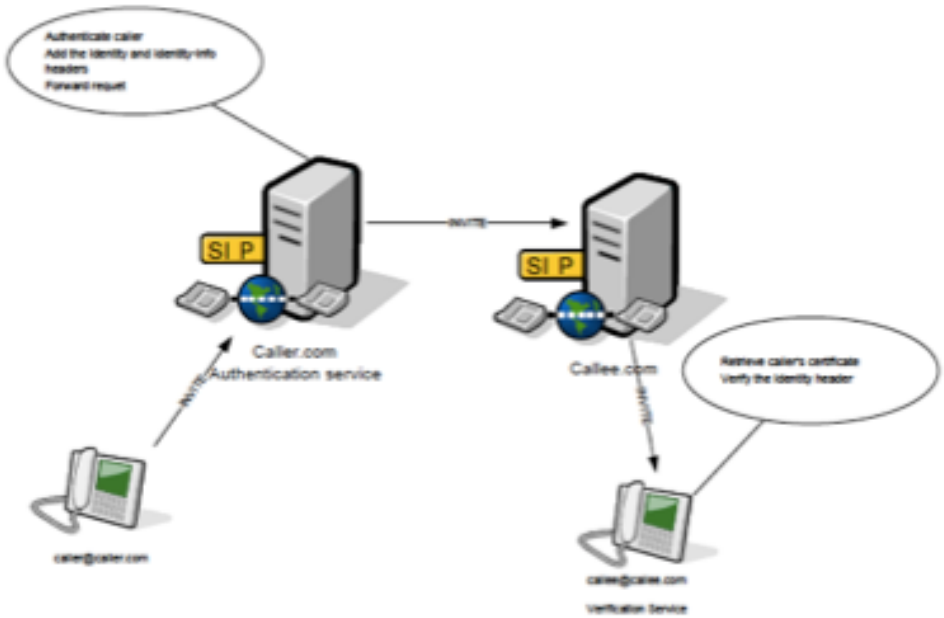

Figure 10 Strong User Identity

As illustrated in *Figure 10* a caller wishing to contact a callee sends the INVITE message to the authentication service. The authentication service authenticates the user using HTTP digest or by checking whether the request arrived over a secure connection that was established during the registration process. After successfully authenticating the user, the authentication server adds two new headers to the SIP message; namely the Identity and Identity-Info headers.

The Identity header is a signed hash of a canonical "identity string" composed of different parts of the SIP message including the caller's and callee's addresses, the information in the Call-Id and Cseq headers and the Contact header and the SDP body.

The authentication server includes in the Identity-Info a URI that de-references to a resource containing the certificate of the authentication service. The callee can use this information to obtain the public key of the caller's operator and use this information to check the authenticity of the request.

As the SBCs change the SDP body and contact information they also break the signature in

the Identity header. An SBC could naturally do the verification on behalf of the callee and then add a new Identity header. But his would mean that the callee will have to have a trust relation with that SBC.

6 Last Words

Since their first introduction over 10 years ago SBCs have considerably gained in scope and capabilities. SBC of the first generation were dedicated devices with often off the shelf hardware that had the sole purpose of establishing a secure border between the subscribers and the operator's PSTN gateways. These SBCs supported mainly NAT traversal and topology hiding. The second generation of SBCs offered a wider range of features including transcoding, support for more complex call flows as well as video communication. Offering features such as DoS and overload prevention, IPSEC support and monitoring and fraud prevention solutions enhanced the security capabilities of SBCs. In addition to the UNI SBCs vendors started offering solutions for NNI and enterprise scenarios as well. To increase the performance and scalability of SBCs vendors started using dedicated hardware and to decompose an SBC into signaling and media control components that communicate with each other using a protocol like MEGACO[18]. This decomposition allows operators to scale the signaling and media handling capabilities of an SBC independently.

We are currently seeing the third generation of SBCs. Vendors are starting to offer SBCs no longer only as a closed box but as a virtual machine that can be installed on the operator's hardware or in a cloud. Further, SBCs are expected to offer open interfaces to enable a smooth integration into the operator's multimedia service infrastructure.

Already the first generations of SBCs supported multiprotocol communication by supporting both SIP and H.323. The next generation of SBCs will enhance this feature by acting as a bridge between the emerging WebRTC implementations, see [19], and SIP. Furthermore, in order to offer improved support for mobile users, SBCs will support integration with Apple and Google notification systems. This would enable mobile devices to remain in sleep mode but still be reachable to the rest of the world.

7 Acronyms

3GPP	3rd Generation Partnership Project
B2BUA	Back to Back User Agent
IMS	IP Multimedia Subsystem
IP	Internet Protocol
ISUP	ISDN User Part
NAT	Network Address Translator
NNI	Network-Network Interface
PBX	Private Exchange
PSTN	Public Switched Telecommunication Network
RTP	Real-Time Transport Protocol
SBC	Session Border Controller
SAP	Session announcement Protocol
SCTP	Stream Control Transport Protocol
SDP	Session Description Protocol
SIP	Session Initiation Protocol
TCP	Transport Control Protocol
TISPAN	Telecommunications and Internet converged Services and Protocols for Advanced Networking
TLS	Transport Level Security
UAC	User Agent Client
UAS	User Agent Server
UDP	User Datagram Protocol
UNI	User-Network Interface
URI	Universal Resource Indicator

VoIP	Voice over IP

8 References

[1] Schulzrinne, H.; Casner, S.; Frederick, R.; Jacobson, V. “RTP: A Transport Protocol for Real-Time Applications (RFC1889)”, IETF, 1996

[2] Handley, Mark; Van Jacobson. "SDP: Session Description Protocol (RFC 2327), IETF, 1998

[3] M. Handley; C. Perkins; E. Whelan. “Session Announcement Protocol (RFC2974)”, IETF, 2000

[4] J. Rosenberg; H. Schulzrinne; G. Camarillo; A. Johnston; J. Peterson; R. Sparks; M. Handley and E. Schooler. “SIP: Session Initiation Protocol (RFC 3261)” IETF, 2002.

[5] Roach, “Session Initiation Protocol (SIP)-Specific Event Notification” RFC 3265, IETF, 2002

[6] J. Hautakorpi, G. Camarillo, R. Penfield, A. Hawrylyshen, M. Bhatia, “Requirements from Session Initiation Protocol (SIP) Session Border Control (SBC) Deployments”, RFC5853, IETF, 2010

[7] J. Rosenberg; R. Mahy; P. Matthews and D. Wing “Session Traversal Utilities for (NAT) (STUN)”, RFC5389, IETF, 2008

[8] J. Rosenberg “Interactive connectivity establishment (ICE): a methodology for network address translator (NAT) traversal for the session initiation protocol (SIP)”. RFC5245, IETF, 2010

[9] D. Willis and B. Hoeneisen “Session Initiation Protocol (SIP) Extension Header Field for Registering Non-Adjacent Contacts” RFC 3327, IETF, 2002

[10] ITU-T 2004 Q.1912.5: Interworking between session initiation protocol (SIP) and bearer independent call control protocol or ISDN user part

[11] TR 180 001: Telecommunications and Internet converged Services and Protocols for Advanced Networking (TISPAN)

[12] 24.228 T 2007 Signalling flows for the IP multimedia call control based on session initiation protocol (SIP) and session description protocol (SDP). Technical specification group core network and terminals, 3GPP.

[13] J. Rosenberg and H. Schulzrinne “Reliability of Provisional Responses in Session Initiation Protocol (SIP)”, RFC 3262, IETF, 2002

[14] S. Dawkins, “IP PBX / Service Provider Interoperability”, SIPconnect 1.1 Technical Recommendation, SIP Forum, 2011

[15] B. Roach, "Registration for Multiple Phone Numbers in the Session Initiation Protocol (SIP)", RFC 6140, IETF, 2011.

[16] Ramsdell “Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.1 Message Specification” RFC 3851, IETF, 2004

[17] J. Peterson and C. Jennings “Enhancements for Authenticated Identity Management in the Session Initiation Protocol (SIP)” RFC 4474, IETF, 2006

[18] C. Groves, M. Pantaleo, T. Anderson, T. Taylor, “Gateway Control Protocol Version 1”, RFC3252, IETF, 2003

[19] H. Alvestrand, “Overview: Real Time Protocols for Brower-based Applications”, draft-ietf-rtcweb-overview-04, 2012

9 About FRAFOS

FRAFOS GmbH is a manufacturer of VoIP solutions with offices in Berlin and Prague. FRAFOS was incorporated as privately held company in May 2010, in Berlin, Germany.

The history of FRAFOS team and technology goes back to the late nineties. As researchers at the prestigious German public R&D institute Fraunhofer FOKUS, the FRAFOS founders were the among the first to work the SIP and RTP standards and to develop open source solutions that paved the way for the VoIP revolution.

FRAFOS offers SIP session management and security solutions of the latest generation that come either as a standalone solution or as a cloud ready implementation. The flagship product of FRAFOS, the ABC SBC, offers open interfaces and built in multimedia applications such as recording and announcements. The ABC SBC enables the operators to simplify their service infrastructure and prepares them for future challenges.