Securing VoIP networks

SAFE CALL

Eavesdropping on conversations on a LAN is easier than ever thanks to insecure VoIP installations. You don't need to bug restaurant booths or tap phone lines – standard Linux tools are all a hacker needs.

BY CHRISTOPH EGGER AND MICHAEL HIRSCHBICHLER

Many small to mid-sized enterprises simply connect their new VoIP systems to existing LANs. Road warriors call in via the Internet, and remote branches use a standard connection to reach the mother ship. Unfortunately, this kind of installation doesn't provide anything in the line of VoIP infrastructure security.

In this article, we look at some of the special concerns affecting VoIP and describe some strategies and optional protocols for protecting voice communications.

THE AUTHORS

Michael Hirschbichler is a Research Associate at the Vienna University of Technology Institute of Broadband Communications (IBK; *http://www.ibk.tuwien.ac.at*). His scientific work focuses on performance evaluation, VoIP security, and, more specifically, IP multimedia subsystem security.

Christoph Egger is also an assistant at the IBK. His research focus includes various VoIP security topics, as well as the simulation and emulation of mobile networks and the IP multimedia subsystem.

A Typical VoIP Connection

Session Initiation Protocol (SIP, RFC 3261) [1] is the most popular open VoIP standard for initiating, negotiating, and managing VoIP connections. In combination with Session Description Protocol (SDP, RFC 4566) [2], which handles audio or video codec negotiation, SIP transmits information about the connection between the calling parties. Once the connection is established, the parties send and receive data using the Realtime Transport Protocol (RTP) [3].

Suppose user A in domain A wants to initiate a VoIP connection with user B in domain B. User A sends an *Invite* request (Figure 1) to its own provider's SIP proxy server. The proxy performs a DNS lookup to ascertain the SIP proxy for domain B, then sends the request to the domain B SIP proxy. The SIP proxy server in domain B checks its own location database for the IP address and port that user B registered with it and forwards the request to this host.

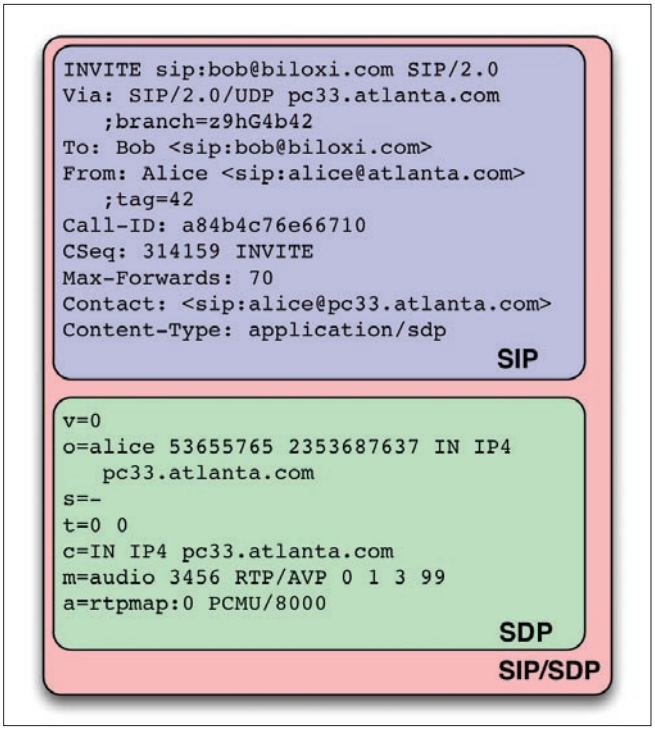

Figure 1: Initiating a VoIP connection – an Invite request comprises SIP and SDP components.

Any response from user agent B, which could be *Ringing* or an *OK* when the recipient accepts the call, is transported back along the same network path.

After setting up the call with SIP/SDP, the system establishes a direct RTP connection between user agent A and user agent B. In some VoIP scenarios, providers use an intermediate media proxy to avoid issues with Network Address Trans-

S/MIME

Secure/Multipurpose Internet Mail Extensions (S/MIME) [7] provide an option for securing the payload only. The client sends the *Invite* request along with an encrypted SDP part to ensure the confidentiality and integrity of the SDP data. Doing so also guarantees that sockets that receive and send the RTP data at the other end really do belong to the authenticated partner. But this only makes sense if the media data are encrypted through the use of secure RTP [8].

Because many SIP proxy servers attempt to rewrite the SDP header, because of the media proxies, encrypted SDP can cause unexpected issues, and working S/MIME-capable clients are unknown.

According to a technique defined in RFC 3893 [9], parts of the SIP header are signed along with SDP. The certificate used for the S/MIME signature can also sign the original From and To headers. The SIP request is divided into three sections for this:

- the SIP request itself is one;
- part two is of the *message/sipfrag* type and contains a copy of one part of the SIP request (To, From, date/time information, and other details);
- part three is the signature for part two.

If the recipient knows the sender's public key, it can validate the relevant data. Implementations of this approach, known as Authenticated Identity Body (AIB), are not widespread.

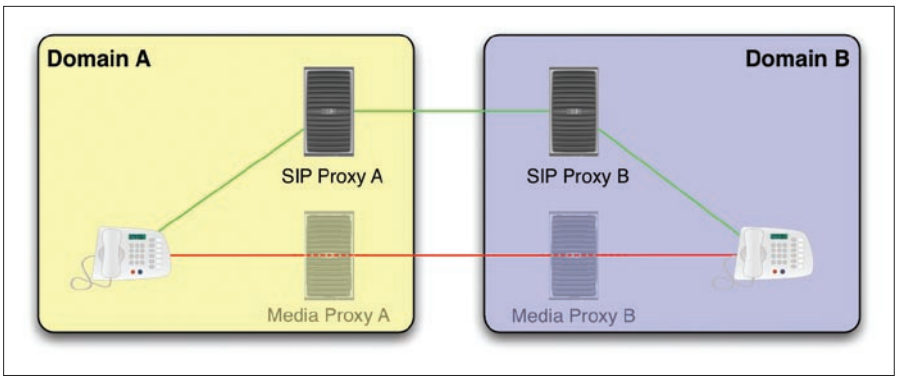

Figure 2: The classic SIP/RTP trapezoid: SIP (green) messages are transmitted via several hops; RTP (red) is typically direct.

lation (NAT) (Figure 2). The advantages of a media proxy are that at least one of the partners does not reside behind a NAT gateway and both terminal devices have counterparts with public IP addresses.

To add the media proxies transparently, your own SIP proxies replace their counterpart's IP addresses in the SDP payload with the media proxy's IP address and port. This convenient technique simplifies the task of getting around NAT devices and firewalls. But at the same time, this approach seriously affects your options for encrypted transmissions. (The topic of NAT traversal is extremely complex, especially in the context of VoIP, SIP, and RTP. More detailed information is available on the web [4].)

What To Do About SIP

Plaintext signaling and its hop-by-hop architecture make SIP very difficult to harden. Because data packets traverse multiple hops, administrators find it difficult to achieve end-to-end security that guarantees availability, confidentiality, and integrity. Each hop en route can add, remove, or manipulate headers. This design makes it impossible to sign the headers.

Signing the SDP payload as an alternative also turns out to be unsatisfactory because SIP includes confidential signaling information in the header. An attacker could, say, manipulate a packet's From header to alter the caller ID, affecting the validity of the payload signature.

The authors of RFC 3261, the current SIP standard, were very much aware of the security issues surrounding SIP, and they responded by adding SIP Secure Schema (SIPS) [5]. A user agent that supports SIPS expects Transport Layer Security (TLS) [6] as the underlying protocol for signaling between two hops to the last SIP proxy server.

TLS-based encryption brings an additional layer of security to SIP. SIPS, however, does not provide a complete solution. The problem with this approach is that user agent A has no way of knowing whether each hop along the path has transmitted the request via a separate TLS-secured connection. The standard *only* defines TLS security to the SIP proxy server in domain B. The connection between the proxy and user agent B is unencrypted.

As an alternative, user agent B can act as a TLS server, but few clients actually implement this technique. Currently, developers are working on a standard that resolves this issue by letting user agent B set up a TLS tunnel to proxy B and keep the tunnel open for incoming requests.

Another problem facing SIP is that, if UDP is used as the transport, it is easy for attackers to replay a request with manipulated headers to hijack the user's identity and make calls. To prevent this kind of attack, most SIP proxy systems support user authentication. Authenticating against your own SIP proxy server provides protection against identity theft and misappropriation of resources. When the user agent registers, or when a call is set up, systems typically use an http digest-based handshake as the authentication algorithm.

Keep in mind, however, that authentication alone does not protect against man-in-the-middle attacks because the integrity of the request cannot be validated. An attacker could sniff authentication information and use different headers to manipulate a call.

Authenticating every single request is not an option in some scenarios; for example, *ACK* and *CANCEL* requests do not expect a response and thus do not support a handshake. The standards support other forms of SIP authentication, but overemphasis on authentication stresses the provider's SIP proxy, in-

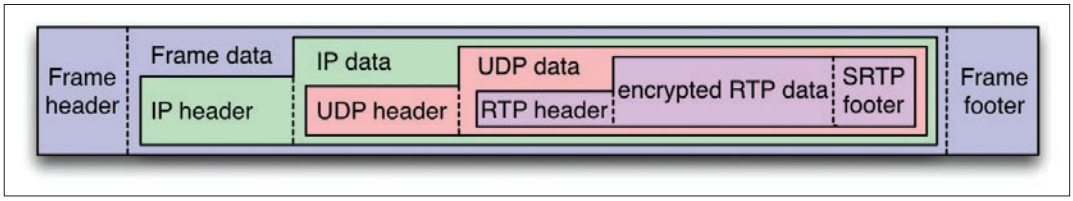

Figure 3: SRTP transparently provides a layer between RTP and transport layers to provide protection against replaying and to ensure privacy, integrity, and authenticity.

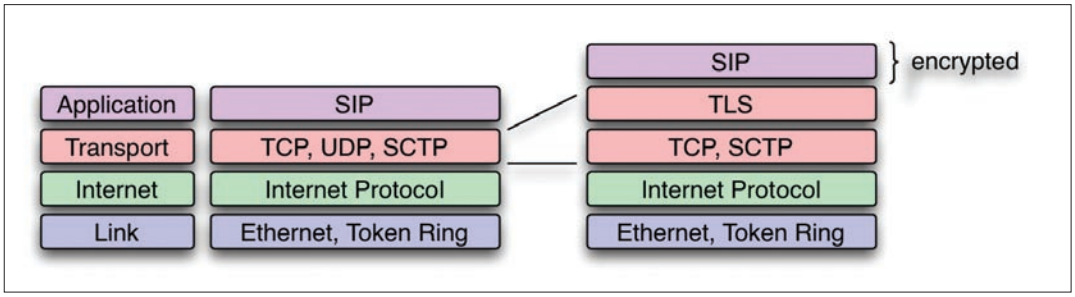

Figure 4: Because TLS is transparent from the application layer's point of view, it offers protocols a universal security mechanism for applications that don't provide their own security.

creasing the delay and affecting connection quality.

What To Do About RTP

RTP does not use encryption and relies on the connectionless UDP as its transmission protocol. This combination makes it simple for an attacker to reroute, sniff, and manipulate data. RFC 3550 [10], which defines RTP, takes this into consideration, providing an encryption option to guarantee data confidentiality.

RFC 3550 points to Secure Real-Time Transport Protocol (SRTP; Figure 3) [8] as a preferred method for extending RTP's functionality through the transport of RTP packets as encrypted payload data. SRTP does not include mechanisms for creating and exchanging keys and thus relies on external methods such as Multimedia Internet Keying (MIKEY, RFC 3830) [11].

The key exchange takes place in the course of the *Invite* dialog. MIKEY offers various approaches, including one that uses preshared keys, in which user agent A encodes the SRTP key to be exchange with a shared secret (MIKEY-PSK). This key exchange only requires a single message; the two SRTP keys for this session can be transmitted with the *Invite* request. MIKEY also supports a key exchange based on a Public Key Infrastructure (MIKEY-RSA), in which the initiator sends user agent B's public key and also sends the session key in the SDP packet to user B.

Another MIKEY method (MIKEY-RSA-R) completely does without a prior exchange of keys or a public key infrastructure. User agent A transmits its own public key, and user agent B responds by generating the SRTP session key and sending it back to user A.

Phil Zimmermann, the inventor of PGP, developed Zimmermann Real-Time Protocol (ZRTP) [12], an alternative key exchange method for initiating an SRTP call. ZRTP does not replace SRTP but extends its functionality.

In contrast to MIKEY, ZRTP does not use the signaling path to transmit the SRTP key information but relies entirely on the media path. An RTP connection is first established and used to exchange the session keys. After voice authentication, the clients switch to encrypted SRTP.

Although end-to-end encryption is possible and the data is protected en route between the user agent and the provider, encrypted communications terminate with the next media proxy and not with the call recipient. This gives the provider network full access to the media content. Just as with proprietary technologies such as Skype, at issue is

Skype and Security

The proprietary VoIP provider Skype confirmed when asked that it does not have an official security policy. An independent survey [13], part of the source code to which the author was given access, confirms that the security algorithms are state of the art and correctly implemented. Despite this, the Skype application remains a black box and is not open to critical expert appraisal. It is impossible to rule out the existence of backdoors or appliances that allowa hackers, or authorities, to eavesdrop and manipulate conversations.

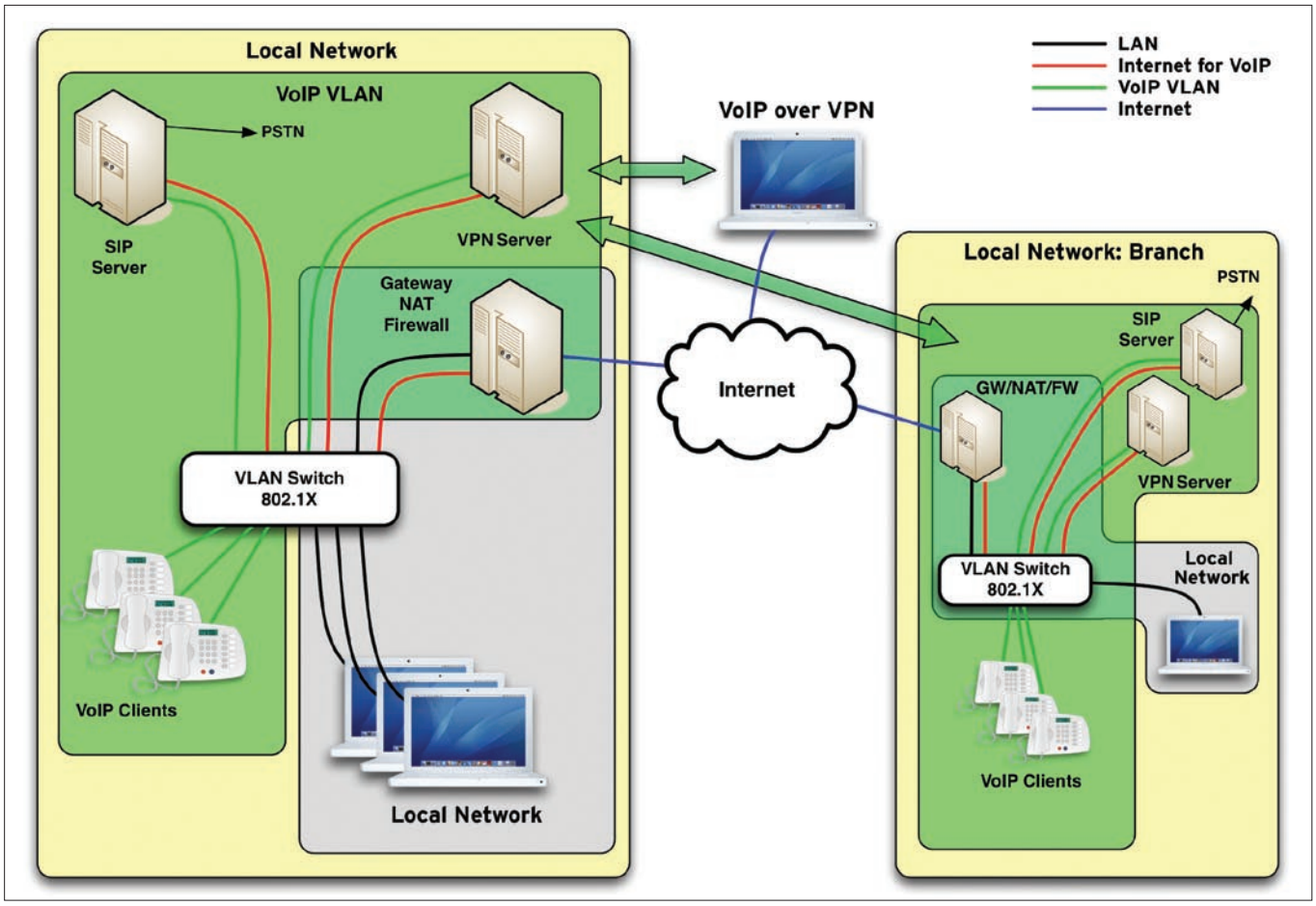

Figure 5: A network with a hardended VoIP system.

TLS and IPSec

Transport Layer Security sits above the transport layer as an enhanced version of SSL (Figure 4). Participants are authenticated through the TLS Handshake Protocol by means of asymmetric cryptography using a public key approach. In the course of mutual authentication, the communication partners negotiate a specially generated, symmetric session key. To verify integrity, they add an SHA- or MD5-encrypted Message Authentication Code (MAC) to the message. For more information on TLS, you can read RFC 4346 [14].

In contrast to TLS, Internet Protocol Security (IPSec) resides directly in the IP layer. Developed in the course of IPv6 standardization, IPSec adds security options to IPv4 and also works transparently for the application layer to provide additional protection to protocols without their own security mechanisms. Because it resides one layer below TLS, it does not need a reliable transport protocol; however, it also supports datagram-based protocols like UDP. IPSec supports two different modes: transport and tunnel.

whether or not the manufacturer is trustworthy (see the "Skype and Security" box).

The Network

Experts agree that one important step in the quest for VoIP security is to separate the VoIP network from the ordinary LAN traffic. The complex and relative insecure nature of computer networks adds many opportunities for eavesdropping and other forms of attack, and seasoned admins are well aware that physical access is more or less identical to a hacked system.

To isolate the VoIP network, use physical separation or even a virtual LAN (VLAN) configuration. Of course, separating VoIP traffic will not protect you against physical access by an attacker who connects to a free port of the voice network. (If a port is accessible, the attacker can simply hitch up a laptop and spoof a phone's MAC address.) The best way to combat this kind of local intrusion is to use additional 802.1x [15] [16] authentication against the switch.

To ensure secure VoIP communications with the enterprise system for road warriors, it makes sense to route the connection via a VPN tunnel. This setup should support low-bandwidth codecs, such as GSM (Global System for Mobile communications) [17]. As an alternative, you might prefer to use SRTP and SIPS-S/MIME, if clients and servers support this option, because of the lower protocol overhead.

Figure 5 shows a combination of the techniques discussed in this article in which the VoIP infrastructure is isolated from the remaining system. The link between the sub-branch and head office uses one or multiple VPN tunnels. (When forecasting the bandwidth, it is important to take the VPN protocol overhead into consideration.) The PSTN (Public Switched Telephone Network) connection can be handled separately at each branch or routed via the head office. A failover link for each branch is a good idea. This keeps your branch offices reachable, even if the IP connection fails or is overloaded.

Conclusion

If you are thinking about adding a voice component to your network presence, it makes sense to plan your approach to VoIP security before you begin. The hardware and software tools of the VoIP environment provide a number of interesting security options. First determine which protocols and components you need for your VoIP network, then shop for tools that provide the necessary support. Table 1 shows the results of our research into the compatibility of phones and VoIP appliances by various manufacturers.

If you already have a VoIP network, simple techniques such as VLAN isolation and strategic use of available encryption alternatives will help you build a better and more secure environment for VoIP communications. ■

Table 1: VoIP Clients and Servers

Manufacturer	TLS	IPSec	SRTP	Comment
Grandstream	✔	✘	✔	Not all devices, status "pending"
Snom	✔	✘	✔	Not all devices, some incompatibilities
Zultys	✘	✘	✔	AES
CrypTone	✘	✔	✘	
Server				
Asterisk	✔ (with patch)	✔ (OS)	✔ (with patch)	
Kamailio (OpenSER)	✔	✔ (OS)	✘	
OpenSIPS (OpenSER)	✔	✔ (OS)	✘	
SER	✔	✔ (OS)	✘	

INFO

[1] SIP standard, RFC 3261: *http://www.ietf.org/rfc/rfc3261.txt*

[2] SDP standard, RFC 4566: *http://www.rfc-editor.org/rfc/rfc4566.txt*

[3] RTP reference: *http://www.voip-info.org/wiki-RTP*

[4] SIP, SDP, RTP, and NAT: *http://swik.net/SIP/del.icio.us+tag%2FSIP/Intruduction+to+SIP%2FSDP%2FRTP+and+NAT/bd1m0*

[5] SIP security: *http://www.cisco.com/en/US/docs/ios/12_3/vvf_c/cisco_ios_sip_security_application_guide/sipsecov.html*

[6] TLS: *http://en.wikipedia.org/wiki/Transport_Layer_Security*

[7] S/MIME: *http://en.wikipedia.org/wiki/S/MIME*

[8] Secure RTP: *http://en.wikipedia.org/wiki/Secure_Real-time_Transport_Protocol*

[9] SIP, RFC 3893: *http://www.ietf.org/rfc/rfc3893.txt*

[10] RTP, RFC 3550: *http://www.ietf.org/rfc/rfc3550.txt*

[11] MIKEY, RFC 3830: *http://www.ietf.org/rfc/rfc3830.txt*

[12] ZRTP: *http://zfoneproject.com/zrtp_ietf.html*

[13] Skype survey: *http://www.anagram.com/berson/skyeval.pdf*

[14] The TLS protocol, RFC 4346: *http://www.ietf.org/rfc/rfc4346.txt*

[15] 802.1x authentication standard: *http://en.wikipedia.org/wiki/802.1x*

[16] 802.1x and attackers on the same port: *http://searchsecurity.techtarget.com/expert/KnowledgebaseAnswer/0,289625,sid14_gci1268965,0.html*

[17] GSM communications: *http://en.wikipedia.org/wiki/Global_System_for_Mobile_Communications*