

Problems With Wiretapping of VoIP Services

Matthew Bates and Thiha Min

INTRODUCTION

We live in an age of increased technological connectivity. Internet Telephony, and specifically Voice over IP (VoIP) stands poised to become more and more prevalent as a primary means of communication. VoIP services have many advantages over traditional public switched telephone networks (PSTNs), chief among them are reduced cost, improved scalability, variety of products/services, increased bandwidth availability, and ease of maintenance. The PSTN (i.e., the circuit switched network) is highly centralized, with traffic passing through a few focal connection points. VoIP traffic on the other hand tends to be decentralized, with the main points of connectivity lying on the fringes of the network – even in the homes of the end users. With the ability to route calls over a dedicated voice network, or even the Internet itself, there exist a myriad of possible paths the data can take. Due to several features of VoIP, like decentralization of the data routes and placing more power in the end users' hands, we must rethink the security strategies that have been regularly applied to PSTNs.

The need for governments to have some measure of appropriate control over civilian communications has been evident for centuries. From intercepting paper mail traffic to eavesdropping on wireless calls, law enforcement agencies have become accustomed to having access to nearly all forms of communication. The earliest recorded case of American communications eavesdropping dates to 1624, when the governor of the Plymouth Colony intercepted mail being sent back to England. Wiretapping started soon after the advent of the telegraph, and was used prominently by the government (both union and confederate) during the Civil War. It was not until 1918 that the first official wiretapping laws were passed. Initially meant for counterespionage purposes, these wiretaps were soon used in the fight against bootlegging and general crime.ⁱ It was only a scant 10 years later that the first wiretap case appeared before the Supreme Court (*Olmstead v. United States, 1928*). When the court ruled in favor of the government, the stage was then set for future wiretapping legislation.ⁱⁱ

Title III of the Omnibus Crime Control and Safe Streets Act of 1968, the Foreign Intelligence Surveillance Act (FISA) of 1978, and the Electronic Communications Privacy Act of 1986 set the modern stage for legislated governmental wiretapping in America. More recently, the 1994 Communications Assistance for Law Enforcement Act (CALEA) and 2004 Joint Petition for Expedited Rulemaking filed by the DOJ, FBI, and DEA have generated much additional discussion concerning wiretapping and the areas to which it applies.

CALEA requires telecommunication carriers to maintain an infrastructure capable of delivering specific conversations and data (such as callee, caller and time) to governmental authorities and to assist in wiretapping when appropriate. Likewise, if the carrier encrypts the communication channels for the end user, it would be their duty to decrypt them at the law enforcement agency's request. We understand how this applies to traditional PSTNs, but how does it apply in the VoIP space?

There are two basic categories into which VoIP communications fall. The first involves a service where some third party ensures calls are connected properly. This service provider is in charge of all intermediate aspects of the connection and may even

supply all necessary equipment. The second category is more user provided (or point-to-point), where both end users download or create compatible telephony software and connect without the involvement of any service provider. There currently exist many examples of both types of approaches – both free and paid (see Table 1 below for a list of free softphones). No matter how it is implemented, a VoIP call is likely to have its data travel over insecure routes (e.g. the Internet) – routes from which interested parties can easily capture data, potentially eavesdropping on the call. It has only been natural that security solutions have arisen which attempt to ensure VoIP privacy.

A number of services, end-to-end software clients, secure connectivity programs, and encryption libraries, both paid and public domain, allow for the easy encryption of Internet telephone calls. In some of these cases, the actual encryption is provided by a third party, while in others it is done by the users themselves. In many cases the conversations can be strongly encrypted with minimal effort. (see Table 2)

CALEA, and its surrounding documents, have engendered thousands of comments (both observatory and official) about wiretapping and VoIP calls. Some of the controversy surrounds the definition of a “telecommunications carrier”. It yet remains to be seen whether VoIP service providers fall under this category. As of yet, no court has ruled that they do, but the 2004 Joint Petition for Expedited Rulemaking requests that VoIP service providers be included under the reach of CALEA. If granted, these providers would then be obligated to ensure that proper governmental agencies are able to wiretap/capture necessary VoIP calls, and to undo any encryption they may provide. This, law enforcement agencies hope, would result in their ability to continue their voice surveillance activities to the same degree in a slightly modified arena.

However we theorize that *much of the legislative discussion surrounding VoIP fails to acknowledge how easily and strongly point to point calls can be encrypted*. With strong end-to-end encryption, it does not matter whether or not somebody can capture the data, in that it will remain unintelligible until decrypted. It is one of our purposes to show that with a few readily available tools, most – if not all – of the current VoIP wiretapping legislation and controversy becomes moot, and that the wiretapping strategies which have been taken for granted may well be disappearing.

SCENARIOS

Let’s take a brief look at what actually happens with the data during an unencrypted VoIP call. The caller and the callee both must have the softphone application running on their computers when the call is initiated. Many programs are designed to “run in the background,” using few resources but still waiting for calls, whenever the machine is on. The calling party must bring up the program to initiate the call. This generally involves typing in the IP address or screen name of the other party and then hitting a ‘dial’ button. Now the calling softphone attempts to establish a connection with the callee. Some data must be exchanged. Signals are sent and received which the softphones understand to represent messages like: ‘Invite’, ‘Busy’, ‘Acknowledge’, ‘Cancel’, ‘Bye’, etc. A formal protocol like SIP, H.323 or SRPT is generally used to exchange these messages, but some programs used their own codes to establish the session. While using a formal protocol allows for greater interoperability, developing your own may allow you to remain within the protocol used to transport data (e.g. sending codes in UDP). Establishing the call may also include additional data like

agreeing which audio codec, transport protocol, or encryption algorithm to use in that case that multiple options are supported.

Once the call has been established, verbal communication commences. A microphone “picks up” the sound waves at either end. An audio codec then digitizes and compresses the sound in some uniform manner. The digitized audio can then be taken from the codec’s output buffer and broken into appropriate size pieces to send in a data packet. A packet (often UDP, RTP, or just IP) is then constructed with the information necessary to reach the other party and sent. Once received, it undergoes a reverse process. The data is extracted from the packet, buffered, decoded by the callee’s codec and played through their speakers or headset as analog audio.

There are several places the data could be encrypted. The important idea is that it must be encrypted and decrypted at the same point in the process. Perhaps the most natural place to encrypt the data is before it is inserted into the IP packet, but it could also be encrypted before being encoded or after being inserted into the packet. If the entire packet is encrypted, then some additional information may be hidden from eavesdroppers. For instance this method could be used in conjunction with IP spoofing (with the true IP address still residing within the higher layer encrypted packet) in an attempt to conceal the identity of the sending parties.

IMPLEMENTATION

There are many existing ways in which to conduct a voice conversation over the Internet (both encrypted and not). We will highlight several specifics and speak briefly of the state of technology of the tools in each of these areas. The main categories we will discuss are: (1) Open-source, encrypted, non-centralized softphones, (2) Closed-source, free, encrypted, non-centralized softphones, (3) Both open and (4) closed-source, free, non-encrypted, non-centralized softphones and (5) Closed source, free, non-encrypted service provided softphone clients. See Table 3 for implementation details and an example in each category. We will not discuss commercial VoIP phones or softphones simply because there is no need to go further than the public domain to get quality results. The state of technology for these paid services (looked at but untested by us) is probably even more robust than of its free counterparts discussed here.

Arguably the best category from which to secure our conversations is that of open-source, non-centralized, encrypted softphones. Software in this category, being open source, can be subjected to independent analysis or modifications to ensure that it meets the security expectations of the users. Being non-centralized (or point-to-point), there is no entity that could be viewed as a “telecommunications carrier” and would have control over the data encryption – and therefore be obligated to decrypt it. Furthermore, with build-in encryption there is no need to use more than one program to ensure that the conversations would be secure.

Of all the categories we looked at, this one had the fewest useable softphones. The most notable example in this category is a program called Speak Freely. Originally started over a decade ago, this open source project supports several types of strong encryption (including 128-256 bit AES, Blowfish, DES, IDEA). We were able to setup a secure VoIP call in both Windows and Linux environments with minimal effort using Speak Freely. It is slightly less user-friendly than some of the other applications we tested in that it only supports bidirectional half-duplex conversations. However, in a

utilitarian sense, a group of users only needs minimal functionality in order to thwart the efforts of governmental authorities who would try to intercept their conversations.

Perhaps the most user-friendly category in which to find software to secure Internet conversations is that of the closed-source (but free), encrypted, non-centralized softphones. As compared with the previous category, applications here boast all the same benefits of being free, encrypted, and non-centralized. The loss of being able to analyze the source code is arguably made up for by the larger variety of more robust programs.

Perhaps the best example in this category is Skype. Skype blurs the non-centralized condition slightly as there is a Skype server which verifies user public keys at login, but the bulk of the service is carried out peer-to-peer. Skype uses 256 bit AES encryption to strongly secure the calls of all 15+ million users. The voice quality is excellent – arguably better than that of traditional telephony. Skype also does not require users to know each others IP addresses as every user has a unique screen name, which they use to connect to each other. It also supports other features like text IM messages and a shared whiteboard, and is available for both Windows and Linux platforms.

Statements like the one made by FCC chairman Michael Powell are particular illustrative of the change to come. “I knew it was over when I downloaded Skype,” Powell explained. “When the inventors of KaZaA are distributing for free a little program that you can use to talk to anybody else, and the quality is fantastic, and it’s free – it’s over. The world will change now inevitably.”

Fortune Magazine, 16th February 2004

Though they alone cannot secure VoIP calls, non-encrypting softphones can still be of use in creating a secure conversation. Most of the phones we looked at were open source and non-centralized (point-to-point). Phones in this category tended to be easy to use and provide cross-platform support. Those applications that are open source also leave open the possibility of incorporating an encryption library into the source to create a secure softphone with minimal new coding.

One open source softphone we worked with was I Hear U (ihu). It uses ‘speex’ audio compression codec and Qt graphical libraries. It is still in its experimental release, so the program occasionally crashed with a segmentation fault. It supports full duplex audio. This program, like most of the others we tried, was tested on Fedora core 1. I Hear U would be a good candidate application to add encryption to later because of its relatively simplicity of design. A good closed-source example we found to work quite well was Sjphone. Though its source code is not open to modification, it can still be incorporated into a system to secure VoIP calls.

There is at least one easy technique that can be used in conjunction with non-encrypting softphones to secure end to end conversations. Tunneling, or port forwarding, is a process which can create a secure ‘tunnel’ in which to forward other data. For example: let’s suppose a softphone communicates on port X between the localhost and remotehost. With port forwarding, a secure link is created between the localhost and remotehost on port Y. All data sent from the softphone to port X on localhost is instead routed through the tunnel and sent out (encrypted) through port Y. The remote host receives the encrypted data on port Y, decrypts it, and forwards it to its own port X. The application never knows the ‘tunnel’ exists, yet the data can be sent securely. In other words, even if the carrier obtains the signal they can’t decrypt it.

Probably the most popular form of port forwarding is through SSL tunneling using programs like Stunnel. Unfortunately, being connection oriented, SSL can only

send TCP data, not the UDP data used by most VoIP applications. A solution is to use yet another program to wrap the UDP data in a TCP layer before sending it.

Alternatively, a few programs exist which do this directly and create their own UDP secure tunnels. Zebedee is an application which can tunnel both TCP and UDP data using the Blowfish algorithm to provide strong encryption. When used in conjunction with a non encrypting application like IHearU or SJphone, an encrypted VoIP call can be established with minimal setup.

The final category we looked at included closed source, free, non-encrypted service provided softphone clients. Programs like NetMeeting and AIM have their connections established by and/or through central servers. Software in this category represent the least likely candidates for encryption, because we cannot access the source code and cannot control the ports/connection. However, it may possible to use this type of software in conjunction with other methods to create a secure call. One idea (which we did not implement) would be to try and encrypt the audio data before it is picked up by the proprietary software (e.g. to encrypt it coming out of the sound card). A far easier idea, and one we did implement, is to route the calls through a Virtual Private Network.

All data traveling between two computers within a Virtual Private Network (VPN) can be both compressed and encrypted. A VPN is very similar to a tunnel in that all data passing between the two virtual IP addresses in indiscriminately encrypted/compressed. There are many ways in which to establish a VPN, but it takes only moments to do the initial configuration between two computers running Windows XP (the server must have XP Pro). After the initial setup is done (using the standard Network Setup Wizard), it takes only two mouse clicks to connect through VPN. Now both the server and client computers are given additional 'virtual IP addresses' to use for communications through the VPN. Also standard in the Windows environment is NetMeeting. Simple start it up, type in the virtual IP address, and hit connect. That is all it takes to establish an encrypted VoIP call. This is one of the strongest arguments we can make for why internet wiretapping laws would be futile – the tools necessary to establish a secure connection - bypassing everything else - are standard, and it only takes a few moments to complete.

In the course of our research, we have concluded that there are several readily available tools and methods with which to create a strongly encrypted internet voice call. Though limited in number now, more of these tools are being created with each passing season. Many of these methods are so basic that any attempt to ban or alter them would profoundly affect the internet as a whole. At this point in time and in the future, *we believe that two end users using public domain tools and minimum setup can effectively create an internet call that cannot be wiretapped.*

Table 1: Examples of free VoIP softphones in some important categories.ⁱⁱⁱ

Name	Point-to-Point (P2P) Service provider (SP)	Encrypted Voice	Open source
Speak Freely	P2P	X	X
Robust Audio Tool	P2P	X	X
Any secure tunneling method	P2P	X	some
Secure Phone Lite	P2P	X	
Foopchat	P2P	X	
Secure Shuttle Transport	P2P	X	
Gaim (with encryption plugin)	P2P	X	
Skype	mostly P2P, but with SP user authentication	X	
I Hear U	P2P		X
CPhone	P2P		X
KPhone	P2P		X
LinPhone	P2P		X
SipXPhone	P2P		X
Gnome-O-Phone	P2P		X
VOCAL	P2P		X
SJPhone	P2P		
X-Lite	P2P		
EZ-Phone	P2P		
PC-Telephone	P2P		
Firefly	P2P		
CDC32	P2P		
LIPZ4	P2P		
XtX	P2P		
INRIA Free Phone	P2P		
MS NetMeeting	SP		
MSN IM	SP		
AIM	SP		
iVisit Lite	SP		
PalTalk	SP		

Table 2: Comparison of encryption key size, number of possible keys, and time required to test all keys for a single key encryption system.^{iv}

Number of Bits (Key Size)	Number of Possible Keys (Key Space)	Time Required (for a home PC)	Time Required (for a supercomputer)
16	65.5 thousand	33.6 ms	negligible
32	4.3 billion	36 min.	2.2 ms
64	1.8×10^{19}	4.5 years	10 hours
128	3.4×10^{38}	5.4×10^{24} years	5.4×10^{18} years
256	1.16×10^{77}	1.9×10^{63} years	1.9×10^{57} years

Table 3: Implementation details for a softphone in each important category.

Name	Point-to-Point or Service provider	Encrypted	Open source	O.S.	Comments
Speak Freely	P2P	AES, DES, IDEA, Blowfish	X	Win/ Linux	Half Duplex, author claims code is messy and easily broken (started in 1991), 128 bit encryption, works.
Skype	mostly P2P, but with SP user authentication	256 bit AES	X	Win/ Linux	Still beta and free, must register for username, tons of users, highly functional.
I Hear U	P2P	none	X	Linux	Simple code, easy to modify, still pre-release so there are a few bugs.
SJPhone	P2P	none		Win/ Linux	Good solid program on both Linux and Windows.
Net Meeting	SP	only for data		Win	Whiteboard, file transfer, and chat are encrypted with the strength and algorithm of MS Internet Explorer.

Endnotes:

ⁱ Death of Privacy in the 21th Century. Simon Garfinkel. Jan 2001. O'Reilly. <http://www.oreilly.com/catalog/dbnationtp/chapter/ch09.html> . Accessed July 16, 2004.

ⁱⁱ <http://www.neighborsforpeace.org/Assault%20on%20Liberty.htm>. Published Nov 21, 2001. Accessed July 16, 2004.

Comment [M1]: We didn't do endnotes for anything on the first draft, so all the notes here are from the 2nd and 3rd draft. We can go back and add them later if it is important.

ⁱⁱⁱ Table 1 is intended to provide the names and characteristics of a small number of VoIP clients. More complete VoIP softphone and services lists can be found at the following web sites:

<http://www.voip-info.org/>

<http://www.pulver.com/>

<http://www.gnu.org/directory/telephony/>

<http://www.voip-news.com/>

<http://www.pernau.at/kd/voip/bookmarks-sip-rtp-ua.html>

<http://www.iptelephony.org/GIP/vendors/client-phones/>

<http://www.voipsupply.com/>

<http://www.epic.org/privacy/tools.html>

^{iv} Snake Oil: Encryption that Sucks. Page: A Thimble-Full of Theory. Accessed July 19, 2004. <http://www.cs.usask.ca/undergrads/dtr467/490/proj/theory.shtml>.