VoIP Fraud: Identifying a Wolf in Sheep's Clothing

Hemant Sengar
VoDaSec Solutions
Fairfax, VA
hsengar09@gmail.com

ABSTRACT

In today's IP telephony world, VoIP service providers and their customers are experiencing a common and rising trend of an attack where hackers compromise legitimate telephone subscriber accounts either from service provider networks, or from one of their customer sites. Once a user account has been compromised, it is used for launching various types of fraudulent activities. Ironically, both users (whose accounts are compromised) and their service providers remain oblivious of any such ongoing fraudulent activities. Generally, such attacks are detected after the fact when damage is already done, either during the call detail records analysis, customer complaints, or billing disputes.

From VoIP service provider's perspective, we ask a fundamental question: *Why does it remain an elusive goal to detect if a call is originating from a compromised user account?* The answer to this question and a feasible solution could be proved as an essential security tool to prevent various VoIP attacks that plague IP telephony world. To this end, we introduce a new dimension to VoIP security, namely *device authorization* along with already existing and widely deployed *user authentication*. The device authorization scheme exploits two unique aspects of calling devices: 1) analog-to-digital conversion process of audio signal; and 2) implementation of SIP timers. By passive and remote observation of signaling and media streams, we establish a relationship between the two and make sure that an *authenticated* telephone subscriber is using an *authorized* device to originate calls.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*

Keywords

Calling Device Authorization; Classification; Fingerprinting

1. INTRODUCTION

Voice over IP (VoIP) telephony is emerging as an alternative to traditional public switched telephone network (PSTN). In contrast to traditional telephone system (where the end devices are dumb), the VoIP architecture pushes intelligence toward the end devices. This flexibility coupled with the growing number of subscribers becomes an attractive and potential target to be abused by hackers. Nowadays SIP scanning attacks have become quite prevalent where SIP-based elements, both in VoIP service provider networks and their customer sites, are regularly scanned to find vulnerable user accounts [12, 27, 26, 10, 5]. The IP-based private branch exchanges (PBXes) and SIP servers are more frequent targets, because hackers expect to find a number of user accounts (i.e., telephone numbers) that do not use SIP authentication at all or if used, but weakly protected. Dainotti et. al. [7] observed Sality botnet scanning the entire IPv4 address space from approximately 3 million distinct IP addresses and trying to discover and compromise VoIP-related infrastructure. The SIP scanning attacks have become so endemic that recently in May, 2013, Acme Packet – the topmost global provider of session border controller (SBC) technology for service providers and enterprises – released a SBC plug-in `sipShield` to prevent SIP scanning attacks from most commonly known tools [18]. The enterprise and service provider voicemail systems are other lucrative targets where hackers try to find voice mailbox numbers protected by default or easily guessable passcodes. Once a user account or mailbox is compromised, it is used to launch various types of malicious and fraudulent activities. One of the most damaging attacks is billing fraud where hackers steal network services by reselling long distance minutes and causing serious revenue leakage to business customers and service providers. A compromised user account or mailbox can easily be configured with *call forwarding* option to some international or premium service number. In the month of January, 2013, there were a number of reported incidents of toll fraud attacks against several small businesses in New York [24]. In August, 2012, some of the Mississippi counties were hit by hackers stealing $\$100,000$ worth of phone calls to central Africa [20]. Within the last quarter of 2012, the Commission for Communications Regulator (ComReg) received 12 report of known cases of hacking [28]. These regular occurrences of fraud attacks victimize both business customers and their service providers [3]. Many of the recently reported incidents of toll fraud can be found here [6]. The latest (year 2011) report published by *Communications Fraud Control Association* (CFCA) puts two topmost telecom industry worldwide frauds losses as $4.96 Billion (USD) because of *compromised PBX / Voicemail systems*, and $4.32 Billion (USD) because of *subscription / identity theft* [1].

Existing solution: In SIP-based IP telephony, it is possible that an attacker could masquerade as another user and originate calls using forged identity. However, the Internet Engineering Task Force (IETF)'s RFC [25] provides *authentication* as a security measure to verify that a SIP request is originating from a legitimate user-agent

CCS'14, November 3–7, 2014, Scottsdale, Arizona, USA.

http://dx.doi.org/10.1145/2660267.2660284 .

(i.e., SIP client). Generally, a SIP server requires a user-agent to authenticate itself before its request can be processed. At the application layer, digest authentication mechanism (using secret sharing of password) is the most commonly deployed method in the industry for subscriber's user-agent to SIP proxy server communication.
Shortcomings of the existing solution: The existing solution works as long as the security credentials of user accounts remain secure. Now assume that somehow a user account has been compromised by a hacker. The compromised password and telephone number can be reused to authenticate any malicious SIP client toward a VoIP service provider network, and originate fraudulent calls.
Other security solutions: Since VoIP service providers cannot distinguish compromised user accounts, as a last resort two other proactive defensive approaches are used: 1) To *minimize* the attack damage, VoIP service providers implement call admission controls (e.g., the maximum number of active sessions, the number of calls initiated within a predefined time interval etc.) on each of the user accounts. These constraints are effective against flooding type attacks, but cannot prevent any fraudulent activities. 2) Time-to-time call detail records (CDRs) are analyzed to discover any anomalous subscriber behavior (i.e., sudden spike in the number of calls to Cuba, Jamaica etc.). However, it should be noted that such fraud is not easily detectable, and when detected it is already too late as significant damage has already been done.

1.1 Motivation

As of today, a user account's credential is associated with user ID such as telephone number only. Therefore, once a user account is compromised, virtually any device type ranging from software-based SIP client running on a general purpose PC (i.e., softphone) to hardphones (such as Cisco, Polycom etc.) can be configured as a legitimate calling device. From service provider's perspective, there is no way to know if a user ID (i.e., telephone number) is moved and security credentials are applied on some other device. Now assume that we have a capability of identifying a remote calling device and consider a *device authorization* scheme where a user ID is bound to its device(s). Toward developing a device authorization scheme, we must address the following questions:
1.) Why should a service provider care about what device is used by a user to originate calls? First, it is more difficult for a hacker to find a device type (i.e., phone vendor and its model number) associated with a user account rather than performing remote scans of SIP-based elements to harvest vulnerable and weakly protected user accounts. Secondly, whenever a user makes any critical changes on his call feature setting using star codes, or calling voicemail access number and then setting *call forwarding* option on voice portal, a service provider can enforce a rule that such changes can be made if a user is calling from his own authorized device. Any calls coming from the PSTN (i.e., calling number is external to the service provider) can check the voicemail, navigate through the voice portal menu, but cannot configure few critical and more vulnerable call feature settings.
2.) If a hacker knows the compromised telephone number then how difficult will it be to find the associated device type? Today's VoIP attacks are blind in nature (i.e., any type of malicious client can access calling services). Although remote, but still we cannot ignore the possibility that a determined hacker can possibly use social engineering, web search (e.g., who owns the compromised telephone number, what types of phones are used in an organization etc.) to find out the calling device type. Even after knowing the device type, still hackers cannot mimic the behavior of that particular physical device attached with a telephone number, because there is a subtle difference between knowing the device type and mimicking the same physical device.
3.) How does a VoIP service provider know what calling device is associated with a user account? The service providers are in an opportune position where customer device information is already available, though largely overlooked until now. For example: 1) during provisioning of a user account, the associated device information is also maintained within the system as a part of SIP interoperability and device management (please refer to Appendix A for the description of VoIP device management system); 2) before the device is shipped to a customer location or during user account activation, few device specific attributes can be learned and recorded.
4.) If device information is already available to service providers then why it is not used for authentication along with user account credentials? SIP RFC [25] specifies `User-Agent` header field for carrying client information as which SIP client is originating that particular request. However, RFC recommends it as a configurable option because revealing the specific software version might allow it to become more vulnerable to attacks against software with known security holes. Even if there is no software version in `User-Agent` field, still it is of little or no real value. *It is because of inherent difficulty to ascertain the true identity of a calling device as text-based SIP messages can easily be manipulated.* Secondly, the session border controller (SBC) located at the edge of an enterprise network may strip it off (being an optional field) from the request before passing it to the next hop SIP element.
5.) Does the binding of a device with telephone number mean calls will not be allowed from any other device? No, not necessarily. For example, if a subscriber has a phone at his office, one at his home, and a softphone on his laptop, he may still be allowed to originate calls if the service provider has provisioned shared call appearance (SCA) service on the subscriber telephone number (i.e., devices are known to service provider). Even if a service provider is offering a *"bring your own phone"* (BYOP) service, the configured device is learned and recorded against the user account during the account activation process. However, calls from unknown devices are blocked and such devices are treated as unauthorized devices.

1.2 Brief Overview

The proposed device authorization scheme consists of two phases: 1) device identification; and 2) device verification. For two party calling scenario where a caller calls a callee, the device identification (DI) module preferably located at the session border controller (SBC) passively monitors both signaling and media streams of each individual *target* subscribers. A target is a successfully authenticated subscriber, but whose device identification profile is yet to be created and verified. As soon as a user's calling device registers and subsequently originates calls, the SIP and media RTP packets are observed to build a device identity. Building a device's identity is a two pronged task: *classification* and *fingerprinting*. The device classification is based on a simple intuition if same class of devices (i.e., same vendor and model number) look similar and have same type of hardware then there should exist some common yet remotely observable attributes that can put it apart from other classes of devices. The proposed DI module studies few acoustic features from payloads of RTP packets to discover the device class which created it. Similarly, to make each calling device distinct within its own class of devices (i.e., device fingerprinting), we notice that each device has its own unique notion of time. A remote fingerprintee can learn this uniqueness by observing SIP packets. Finally, in the device verification phase, the DI module queries (using API calls) the SIP server where this particular user is provisioned and finds out what device is associated with this user account. The derived device class is compared with the actual de-

vice type assigned to the user. Similarly, the derived fingerprint is also compared with a local existing (i.e., at the SBC itself) device fingerprint. This reference device fingerprint is created during the user account activation i.e., at the very first time when device attaches itself to the service provider network. Now assume that a derived device identity does not match, the established call is interrupted and an alarm is raised so that new password can be assigned to this particular user account. Being a part of device management system, reassignment of new authentication password and then resetting of the device (to bring the changes to the device) can be done at any time without the telephone user's knowledge.

1.3 Contributions

In this paper we develop a device identification scheme encompassing both *classification* and *fingerprinting* techniques. As a part of device classification technique, we introduce a novel idea of analyzing the "sound of silence" from payloads of RTP stream. The silence carried in payloads reveals the device information which created it. Kohno et. al.'s remote physical device fingerprinting is based on clock skew measurements by remote observation of TCP/ICMP packets' timestamps [16]. If a device's packet stream does not contain any explicit timestamp, could we still measure its clock skew? We observe that SIP messages from a calling device can reveal its clock skew without requiring explicit timestamps. We conduct a series of experiments covering a broad spectrum of VoIP service offerings involving SIP-based hardphones, and softphones etc. In all these cases, calling devices are classified within 30 seconds of call establishment. Whereas, device fingerprinting requires ≈ 5 minutes of its presence (i.e., device remains registered) to service provider network. This promising and encouraging results lead to a practical device authorization scheme suitable for VoIP security deployment in the near future.

The remainder of the paper is structured as follows. Section 2 describes the threat model. In Section 3, we revisit already known remote physical device fingerprinting technique. Section 4 discusses few acoustic features that can be analyzed from a voice stream of a caller to learn about device information which created it. Section 5 introduces a new timer-based remote device fingerprinting technique. Section 6 discusses the possible deployment locations of proposed DI module, the basic requirements of DI module, and how the prototype is implemented. Section 7 presents our experimental results. Section 8 surveys related work. Finally, Section 9 concludes the paper.

2. THE THREAT MODEL

We discuss two most common and frequent attacks occurring against VoIP elements of service providers and enterprise networks. 1) Recently, there have been many reported incidents where SIP scanners run around the clock against enterprise PBXes (at the service provider's customer sites) and VoIP service providers' SIP servers itself to look for weak or unprotected subscriber accounts. Generally, such attacks start with finding an IP address where standard SIP port 5060 is open. Next, the SIP `REGISTER` messages are used to scan the SIP element to find an existence of a telephone number. For example, `404 User Not found` and `401 Authorization Required` response messages for a specific telephone number are explicit enough to know whether that number exists within the target system. Once a number is found, a brute force attack follows to guess the credentials attached with the telephone number.
2) The voicemail systems present another venue for hackers to compromise a user's voice mailbox. It is a common practice that enterprises and VoIP service providers publish their voicemail's common access number – a number that can be called from anywhere to access voice portal. The voice portal is an interactive voice response (IVR) application that can be used to manage calling services and voice mailbox, or to change user passcode etc. Generally, the mailbox ID is an individual user's telephone number, whereas the passcode is selected by the users themselves. However, as a common practice, users select these passcodes as easily guessable last 4 digits of the telephone number itself, four zeros, four ones, rows of telephone keypads, sequence numbers such as 1234 etc. In this particular attack scenario, there is no need to compromise SIP authentication credentials. A malicious user can call the voicemail common access number and then starts probing mailbox IDs. The service provider receives such mailbox probing calls as any other normal phone calls (please refer to Appendix B for the description of VoIP architecture). If a hacker succeeds to find a mailbox ID and its passcode then as a next step, hacker exploits voicemail system's outdialing capability. One such vulnerable feature is "*call forwarding always*" where the hacker can provision an international number using phone keypad. Any calls coming to the hacked mailbox number will be automatically forwarded to the provisioned number.

3. REMOTE DEVICE FINGERPRINTING

Kohno et. al. [16] are the first researchers we are aware of who present a remote physical device fingerprinting method for PCs and servers based on clock skew derived by observing TCP/ICMP timestamps. Following the same approach, we use RTP timestamps to fingerprint a remote calling device. The calling device's clock precision affects audio sampling that in turn affects RTP packet intervals. For example, the 160 samples payloads are treated as 20 ms frames. Any error in clock frequency will cause deviation in sampling, and it will affect the 20 ms packet interval. Kohno et. al.'s work is seminal. Although, there are many limitations in applying their approach directly to the calling devices and towards development of a device authorization scheme.

First, let us consider a case where phones are indistinguishable with respect to their clock skew values. The `G.711`, an ITU-T standard samples audio signals at the rate of $8,000$ samples per second with the tolerance of ± 50 parts per million (ppm) [13]. Since most of the calling devices in the market maintain this tolerance range, it is natural to find many calling devices with very close values of clock skew measurements. It allows the possibility of one phone mimicking the behavior of another phone. *How could we resolve this dilemma? Is there a way to know which calling device (i.e., vendor and model number) has actually originated a particular RTP stream?* We developed a highly accurate new device classification technique that can label a remote calling device with its manufacturer name and model number as described in Section 4.

Secondly, compared to PCs and servers, fingerprinting remote calling devices based on RTP timestamps is a challenging job. This is mainly due to the following reasons: 1) being an interactive application, telephones have intelligence and capability to adjust packet's departure time; 2) the calling device's access location cannot be assumed to be fixed (i.e., a case of nomadic subscriber), and hence its network path (along with broadband access method and access router) may change; 3) the network path dependence becomes more prominent due to high packet rate (i.e., 50 packets per sec. for `G.711` codec with 20 ms payload content). Each of the access locations (i.e., access network path) introduces an inherent noise in offset data points differently and independently of each other (please refer to Appendix C for further details). Therefore, measuring device's clock skew from RTP timestamps is unreliable and cannot fingerprint it uniquely and distinctly. We need a more reliable fingerprinting technique. Thirdly, we cannot ignore the possibility that a hacker could spoof the timestamps of RTP pack-

ets. For example, a malicious softphone can be bundled along with the spoofed timestamps. This presents a new challenge: *How could a fingerprintee refer the time notion maintained by a remote calling device (and hence measure its clock skew) without relying on packet timestamps?* We developed a new and more reliable timer-based (in contrast to well-known timestamp-based) remote device fingerprinting method as described in Section 5.

For a device authorization scheme to be a practical and viable solution, we need a methodology that can assign a unique identity to each of the calling devices as unambiguously as possible. To *identify* a remote calling device, we use multilayered approach. First, we classify and label a device at a fine granular level capturing phone vendor, model number, and possibly the manufacturing batch number. Secondly, the calling device is fingerprinted to create a unique identity within its own class. All of these tasks rely on passive observation of signaling and media streams, and hence the proposed device identification technique remains completely undetectable to callers.

4. LABELING A CALLING DEVICE

In this section, our focus will be on labeling (i.e., classification) of remote calling devices. We discuss how the analysis of RTP payloads reveals the device information which created it.

As a person speaks over the microphone, the captured analog audio signal goes through many steps before digitized audio is put into packets for transportation.This analog-to-digital (A/D) audio conversion process taints the media stream with few device specific attributes (related to hardware, software, or the combination of both). Furthermore, we also observe that the effect of device attributes on acoustic features become more prominent when speaker pauses during the conversation i.e., within speaker's silence segments. As of present, `G.711` is the most commonly used codec in Voice over Internet Protocol (VoIP) network. It is also a default codec choice for private branch exchange (PBX), as well as for the public switched telephone network (PSTN). Consequently, in our analysis the caller's audio stream is encoded by `G.711` audio codec. The `G.711` has two variants, the μ-law codec is used in North America and Japan, while the A-law codec is more common in the rest of the world. However, our analysis pertains to both algorithms equally well. At the proposed device identification module, the `G.711` encoded payload of RTP packets are extracted and decoded to get audio samples. These samples are studied across a wide range of acoustic features such as *zero-crossing* rate, *energy*, *samples distribution*, *DC offset*, *frequency* analysis, and *dithering* etc. to infer device information which created it. Within this paper we discuss the following three features only i.e., 1) *silence energy*; 2) *DC offset*; and 3) *dithering* pattern in more detail.

4.1 Silence Energy

The output noise energy is determined by the bandwidth of a filter e.g., the noise energy is dependent upon the spectral response of a filter and one with narrowest bandwidth removes the most noise. However, if we compare a software versus a hardware phone, we find that implementing *digital filters* are more expensive compared to *analog filters*; secondly, low noise performance can better be achieved with filters implemented with analog circuitry. Since the filter design and implementation is same across the same class of devices, we expect these devices to observe and treat ambient background noise level similarly. Therefore, the silence energy could be used as a phone specific attribute for device classification. The caller's utterance i.e., both voiced and silence segments are carried within the RTP stream. We distinguish silence carrying RTP packets (discussed later in Section 6) from the voiced packets. Now

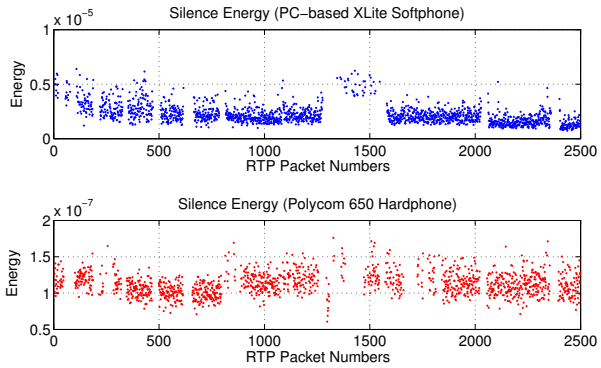


Figure 1: Per Packet Silence Energy (Within RTP Stream)

let's assume that a i^{th} silence packet's payload has N audio samples $x_i(n), n = 1,2,.....,N$. The i^{th} packet's silence energy is calculated as: $E(i) = \frac{1}{N}\sum_{n=1}^{N}|x_i(n)|^2$.

As an example, Figure 1 shows the estimated silence energy at the individual packet level (i.e., 160 samples per packet) for `G.711` μ-law encoded RTP audio streams. The audio streams are originating from two different sources: 1) at the top, `X-Lite` - a Windows-based softphone; 2) at the bottom, `Polycom SoundPoint IP 650` [1] - a hardphone. The X-Lite softphone's silence energy is an order of magnitude higher than the Polycom hardphone.

4.2 DC Offset

The analog voice input signal for the phone's digital signal processor must be amplified with analog circuits prior to analog-to-digital (A/D) conversion, it is most likely that the captured signal carries an unwanted DC component. It means in the presence of DC offset, the mean amplitude of the waveform may deviate from the zero either in positive or negative direction. Using a high-pass filter, DC offset can be reduced in real-time. However, the filter design and its use is very specific to the phone type and its hardware. As an example, let us assume that a user initiates a call from his `Polycom SoundPoint IP 650` hardphone. The `G.711` $\mu-$law encoded payloads from the incoming audio stream is decoded and the first $80,000$ samples (10 seconds) of caller's utterance saying *"Hello!..Hello!"* is shown in Figure 2 (top). Looking into the wave-

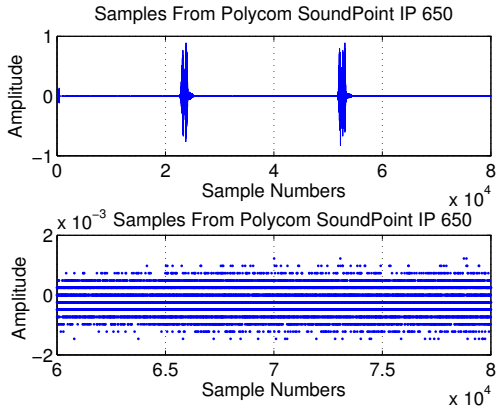


Figure 2: Presence Of DC Offset

[1]Polycom is the global leader in open standards-based unified communications (UC) with 32% global market share.

form, we may doubt if there is really any DC offset component present in the caller's audio stream. However, if we look more closely, especially in the caller's silence, Figure 2 (bottom) shows how the sample values are shifted below zero toward negative values. It shows the presence of negative DC offset within `Polycom SoundPoint IP 650` phone originated audio stream.

Now we discuss how to estimate the presence of DC offset and assign a numerical value to it. Let's assume that within a particular silence RTP packet i, we observe N audio sample values $x_i(n)$, where $n = 1....N$. To estimate the DC offset, we compute the average of all sample values: $DC_i = \frac{1}{N}\sum_{n=1}^{N} x_i(n)$. Assuming that we analyzed I silence RTP packets, the DC offset value for the phone is calculated as the mean of individual DC offset values estimated per packet.

4.3 Dithering Process

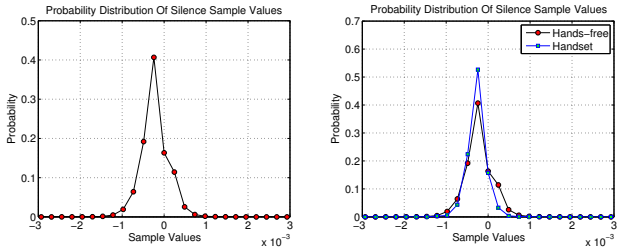
An analog signal is continuous. When an analog signal is *quantized* within PCM digital system, the amplitude of the output signal is limited to a set of fixed values. If a signal is quantized without dithering, there will be quantization distortion related to the original input signal. In order to prevent this, the signal is dithered, a process that mathematically removes the harmonics or other highly undesirable distortions entirely, and replaces it with a constant, fixed noise level [11]. Therefore, adding dithering noise to an analog signal before analog-to-digital conversion prevents digitized signal becoming stuck on one particular value. There are several algorithms in the market from many manufacturers for adding unique dither to audio. For example, `POW-r`, `DitherCD`, `UV22`, and `IDR` are all examples of these types of algorithms. In our analysis, we do not try to discover which algorithm is used to dither audio signal; instead, we study the audio samples distribution within silence segments. We observe that each phone type has their own characteristic silence samples distribution. For example, the silence samples of `Polycom SoundPoint IP 650` phone are binned into 25 bins corresponding to 25 discrete values lying within a sample range of $-3*10^{-3}$ and $3*10^{-3}$. The probability distribution of silence samples is plotted in Figure 3 (a.).

4.4 Handset vs. Hands-Free Talking Mode

Most desk phones in business environment provide two modes of talking – hands-free and handset. In the hands-free mode, a speaker speaks in front of the phone; whereas, in the handset mode, a speaker picks up the handset like a traditional phone. Both internal (on the phone body) and external (on the handset) microphones have different interfaces, we asked ourselves the effect it has on acoustic features. For example, when a `Polycom SoundPoint IP 650` phone is used by the same speaker in both hands-free and handset modes, we observe noticeable differences. Though, the silence energy is still mostly confined within a similar range $1*10^{-7} - 2*10^{-7}$; however, the DC Offset component becomes more negative shifting from $-2.24*10^{-4}$ to $-2.69*10^{-4}$ when the speaker switched from hands-free to handset mode. The silence sample distribution also depends upon the talking mode. For example, as shown in Figure 3 (b.), the silence samples filling a particular bin (with the value of 0.0002441406250) is much higher if a caller is using the hands-free mode.

5. FINGERPRINTING A CALLING DEVICE

Once a calling device is classified and labeled, the next logical question is: *how do we know if an observed voice stream is really from the same authorized device?* For example, it could be argued that since a hacker knows the telephone number of a compromised user account; though remote, but he could possibly be able to discover the associated calling device type too (i.e., phone vendor and model number), and then register a similar phone with compromised credentials. To address this legitimate concern we develop a device fingerprinting technique using SIP packets.



(a.) **Samples Distribution** (b.) **Hands-free vs. Handset**

Figure 3: Silence Samples Distribution

While studying SIP protocol behavior and its implementation across telephony devices, we notice that the device registration process can be exploited at many levels for not only identifying a rogue device, but building a robust device profile.

Registration Process: As soon as a calling device is powered on, it tries to register its current location to a registrar located within the VoIP service provider network. The SIP client sends a `REGISTER` message to the SBC, and being back-to-back user agent (B2BUA) SBC forwards it to the registrar. Both SBC and registrar store the binding information of the user and his current location contained within the `REGISTER` message. The *'Expire'* field within the

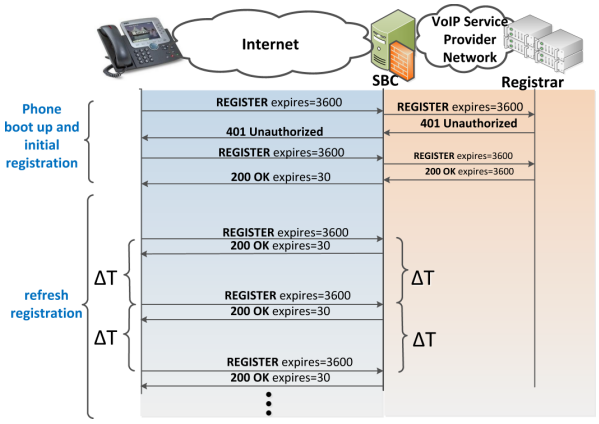


Figure 4: `REGISTER` Message Flow

`REGISTER` message reflects the SIP client's desire for a time duration to which registration should remain valid. The successful registration is acknowledged by sending a `200 OK` response message. This response message contains an *'expire'* parameter indicating the validity interval of the registration. Now it is the client's responsibility to refresh and send another `REGISTER` request before the earlier binding expires. *However, RFC [25] leaves it to the client as when this refresh `REGISTER` should be sent before current binding expires.*

NAT Traversal: Generally, phones are deployed behind a firewall/NAT device. It is the SBC's responsibility to make sure that the firewall does not close the UDP ports[2]. Otherwise, the phone cannot receive any calls. Due to this reason, all SBCs in the mar-

[2]SIP can run over both TCP and UDP. However, VoIP service providers prefer UDP, because of its better performance.

ket have a capability to detect whether a phone is behind a NAT, and if it is, then it forces the client to register more often. The frequent REGISTER messages from a client forces the firewall to keep the UDP port open[3]. As a common practice, the typical value of the *'expires'* parameter is set to 30 seconds by the SBC in 200 OK response message sent back to the client.

Exploiting REGISTER Messages: As shown in Figure 4, even though SBC sends a 200 OK response with registration validity of 30 seconds, the client still sends a refresh REGISTER message much earlier before the actual expiration. *How much earlier does the client send the REGISTER message?* It is dependent upon the values of various parameters that are set in the device configuration file[4]. For Polycom phones, the refresh REGISTER message flow is controlled by the following two parameters: 1) *voIpProt.server.x.expires*, and 2) *voIpProt.server .x.expires.overlap* of the *sip.cfg* device file. Similarly, other phones in the market have their own set of parameters that determine when to send refresh REGISTER messages[5]. By observing the arrival times of refresh REGISTER messages, we could check whether the calling device is behaving as per device configuration set by the service provider. Let us assume an extreme case where device configuration files use default parameter values, and somehow a hacker also uses the same default settings. Now we discuss REGISTER timer-based clock skew measurement which is both difficult to guess and spoof by a remote hacker.

If a device sets its refresh timer for $\triangle T$ time period then the next refresh REGISTER message will be send after the $\triangle T$ timer expires. The same settings of device configuration file forces all devices (of same class) to behave in a similar fashion. However, the device clock used in deriving the $\triangle T$ timer value differs from device to device. To build a device fingerprint, the device identification module records the arrival times of refresh REGISTER messages and estimates the device's clock skew. More formally, let us assume that for a particular telephone device $\mathbb{A}$, t_i is the time when an i^{th} refresh REGISTER packet is recorded by the identification module. We define x_i (i.e., $x_i = t_i - t_1$) as the time elapsed between the first and the i^{th} packet observed by the identification module. Similarly, the w_i (i.e., $w_i = (i-1) * \{\triangle T\}$) is the time elapsed between the first and i^{th} refresh REGISTER packet of the calling device as derived from phone's REGISTER refresh timer value. Now taking device identification module's clock as a reference, we derive *offset* data points. The y_i (i.e., $y_i = w_i - x_i$) is clock offset of the i^{th} REGISTER packet. It gives a set of clock offset data points $(x_i; y_i)$ corresponding to device $\mathbb{A}$. We observe a linear pattern of offset data points. The derivative of the offset with respect to time i.e., skew acts as a fingerprint of the corresponding registering device. We use least square fitting (LSF) to estimate the clock skew from device's offset data points. Given a set of offset data points $(x_i; y_i)$, LSF finds a line $m * x + c$, where m is the slope of the line and c is the y-axis intercept, such that, $\sum_{n=1}^{N} [y_i - (m * x_i + c)]^2$ remains a minimum.

6. EXPERIMENTAL METHODOLOGY

From the service provider's perspective, we discuss the possible deployment locations of proposed device identification (DI) module; what are the basic requirements of DI module to work properly; how our experiments were conducted; and finally, how we prototyped.

Placement Of Device Identification Module: Although, the DI module could exist as an independent device, in a practical deployment scenario it could be collocated with SBC and possibly at the media server (MS). Generally, SBC is placed at the edge of the service provider network representing the access point for its own subscribers exerting control over signaling and media streams. At this location, the DI module can detect if the subscribers are using their own authorized calling devices to access SIP server resources. The MS is another location where both internal and external callers (i.e., callers whose telephone numbers do not belong to the service provider) leave voice messages or access mailbox voice portal menus. Here, the DI module checks if a call coming from a PSTN gateway (i.e., an external caller) is trying to configure voice portal settings related to its outdialing capability.

Requirements Of Device Identification Module: Starting with the device registration, and subsequent call requests, the DI module's only requirement is its ability to observe both to-and-fro SIP signaling messages, and device's media stream. For example, from the SIP call request (i.e., INVITE) and its subsequent 200 OK and ACK messages, the DI module derives three pieces of information: first, the call request is originating from an authenticated subscriber; secondly, the From header field of call request contains SIP URI (i.e., identification) of the caller who is originating this call request; and finally, the session description contained in the INVITE's message body carries media information. The connection information field (i.e., c=) contains media connection information such as media's source IP address that will be sending the media packets. Similarly, the media information field (i.e., m=) contains media type and the port number. From the SDP portion of 200 OK message, we know what codec is negotiated between peers. Based on the collected information, the DI module knows from where to expect the media stream to arrive for a particular user ID, and also how to decode it (based on negotiated codec information).

Experimental Setup: To simulate realistic "real-world" calling scenarios, all of our experiments are conducted using network resources of one of the topmost VoIP service providers in the USA comprising of carrier grade VoIP and network elements. The SIP-based test phones are located in various parts of the country (Aldie in Virginia; Conroe in Texas; Montclair in New Jersey, Downey in California). The phones register through the same access SBC located in Greenville, South Carolina. As a part of CALEA (i.e., wiretapping) compliance requirements, there are no direct communications between two endpoints (i.e., phones), and the SIP signaling and media streams flow through the SBC. Using switch port mirroring, SBC traffic is copied and send to a VoIP probe where we perform our device identification analysis.

Ambient Environment: In our experiments, the audio attributes such as energy, samples pattern, and DC offset are derived from silence samples. However, this silence represents a normal (noisy) office environment. To measure the ambient silence sound level during our experiments, we use an A-weighted sound level meter. The Lanman et. al. [17] sound meter implementation allows the user to acquire samples from the sound card in real-time. A Fast Fourier Transform algorithm is used to estimate the frequency spectrum, and average signal energy is estimated using Parseval's relation. During our experiments, the measured silence signal level is found to vary between $\approx 50 - 55$ dBA (A-weighted decibels).

Implementation Of Device Identification Module: To analyze media payload data, and select a particular RTP packet stream from callers, we develop a Java-based application relying on open source tools such as Jpcap [14] to capture packets from network interface

[3]As a configuration option, SBC can force even non-natted devices to register more frequently.

[4]This password protected device configuration file is a part of service provider's device management system, and the customer (or telephone subscriber) cannot tweak its settings.

[5]The phones using SIP over TCP can also be forced to register more often.

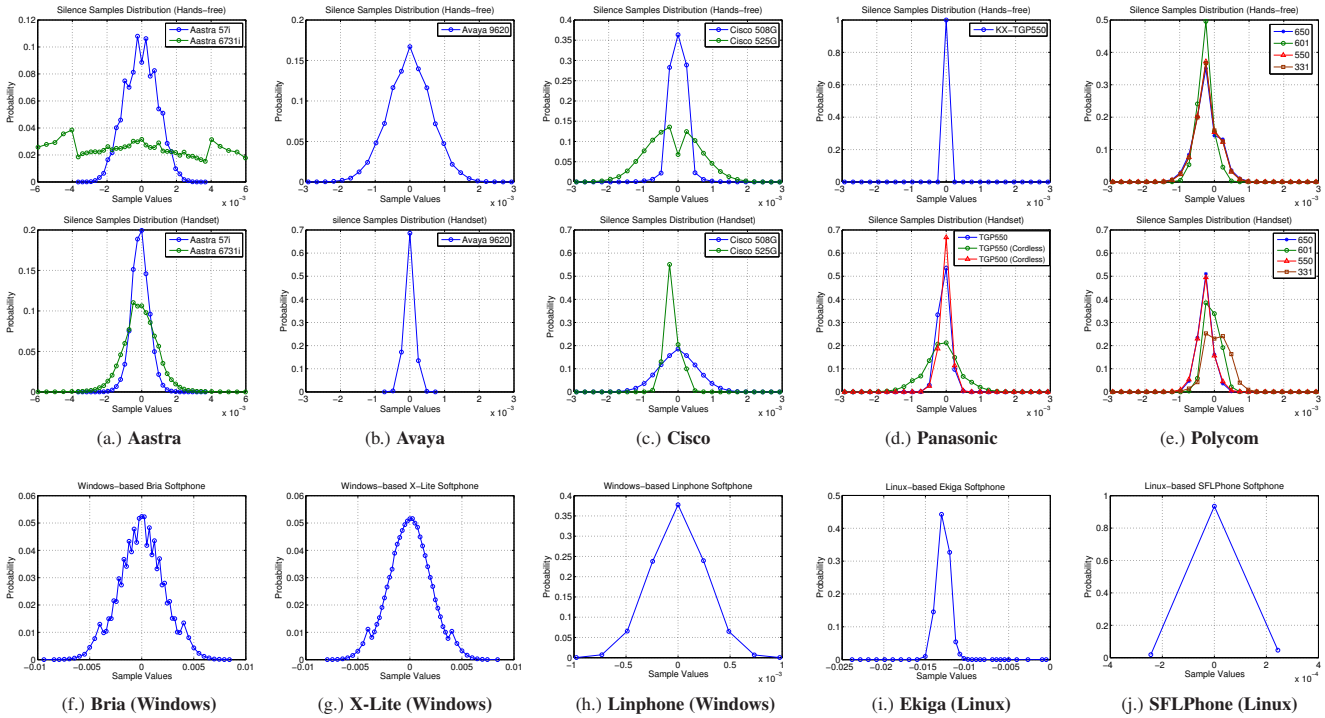


Figure 5: Distribution Of Silence Samples (Hardphones, Windows And Linux-based Softphones)

and Peers [21] to parse SIP messages and decode RTP payloads. The receipt time of packets are recorded to measure the clock skew. For media handling, Peers implements `Incoming RtpReader` and `CaptureRtpSender` as the two main classes. The `IncomingRtpReader` is responsible for RTP depacketization, media decompression, and media playback; however, for media processing, the whole media package relies on standard *Sun Java Sound API*. The Peer's `SoundManager` class implements all its interaction with the Java Sound API. Throughout our implementation, we assume the following audio format:

```
// linear PCM 8kHz, 16 bits, mono, signed, little endian
audioFormat = new AudioFormat(8000, 16, 1, true, false);
```

Within the `SoundManager` class, we implemented audio analysis algorithms for decoded audio samples of each packet's payload. To separate silence packets from the media stream for a particular subscriber, we observe RTP packets for the initial 30 seconds (assuming this time duration is good enough to contain few seconds of silence segments), and decode packet's payload to calculate its energy. Based on the energy level, these packets are binned into their corresponding energy level bins. We have a set of 21 energy bins of small incremental differences. For example, we create 20 equidistant bins for energy between 10^{-9} and 10^{-4}, and one default bin for energy level lower than 10^{-9}. Other higher energy packets are discarded. At the end of observation period, we search for the lowest energy level bin that contains most of the packets. It gives us an energy range where the silence energy is concentrated. The payloads of packets in that particular bin are used to derive silence sample distribution and also to estimate the DC Offset component.

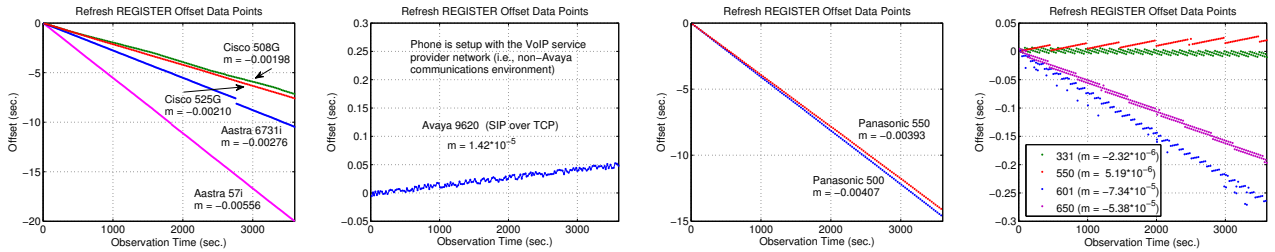
7. EXPERIMENTAL EVALUATION

In this section, we evaluate the effectiveness of our approach for classification and fingerprinting of a remote calling device.

7.1 Classification

In the first set of experiments, we selected 11 different hardphone models from 5 leading phone vendors (namely, Aastra, Avaya, Cisco, Panasonic and Polycom) and 5 of the most popular commercial and open source softphones (Windows-based X-Lite, Bria, Linphone, and Linux-based Ekiga, SFLphone). The softphones are installed on a dual boot laptop computer (2.26 GHz Intel Core2Duo, and 3 Gbytes of RAM) running both Windows Vista and Ubuntu 12.10 linux OS. All of the phones register from one access location Aldie, Virginia (using residential broadband connection) to the SBC located in Greenville, South Carolina. The service provider's SBC is at 13 hops away with 38 ms average round trip time. From each individual phone we made $5-8$ calls and the average values of DC Offset and Silence Energy is tabulated in Table 1. The silence RTP packets collected from initial 30 seconds of phone conversations are decoded and the distribution of silence samples is plotted in Figure 5.

Our experimental results demonstrate that it is possible to extract the device information such as vendor and model number by analyzing the acoustic features from audio payloads of the media stream. For Windows and Linux softclients, although the hardware resources are common, we still see "silence" is not created in the same way. As shown in Figure 5 (f.) and (g.), the X-Lite (free version) and Bria (commercial version) soft clients developed by the same company, use different dithering process, and therefore affect DC Offset values. For hardphones, a careful analysis of the experimental results reveals that the built-in microphone (i.e., hands-free mode) of Polycom phone models such as `SoundPoint IP 650`, `550`, and `331` behave in almost the same manner, and therefore may be using the same hardware. Whereas, the audio from handset microphone of `SoundPoint IP 331` is quite different from the other two models. We also observe that audio processing in `SoundPoint IP 650`, and `550` are identical, in both handset and hands-free modes. The only difference between these two



(a.) **Aastra and Cisco Phones** (b.) **Avaya Phone** (c.) **Panasonic Phones** (d.) **Polycom Phones**

Figure 6: Measuring Clock Skew From Refresh `REGISTER` Messages (Various Hardphone Vendors/Models)

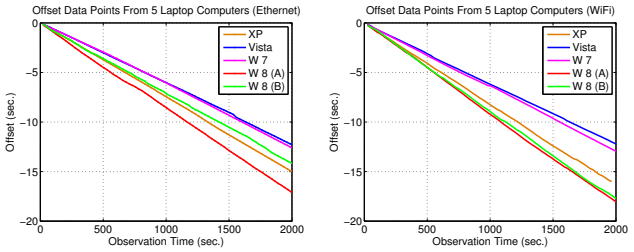
Table 1: Analysis Of Various VoIP Soft- And Hardphones

		DC Offset		Silence Energy	
Phone	Phone	handset	hands-free	handset	hands-free
Vendor	Model	$*10^{-4}$	$*10^{-4}$	$*10^{-7}$	$*10^{-7}$
Aastra	***57i***	−0.85	−0.59	2.24	13.03
	6731i	−0.73	−4.44	1.90	298.1
Avaya	***9620***	−0.098	−0.086	0.19	4.10
Cisco	***SPA 508G***	−0.001	−0.006	0.59	0.55
	SPA 525G	−1.84	−1.54	0.57	2.30
Panasonic	***KX-TGP500***	$-0.29^{\dagger}$	‡	$0.30^{\dagger}$	‡
	KX-TGP550	$-0.74/-1.38^{\dagger}$	0.00	$0.45/3.00^{\dagger}$	0.00
	†With Cordless Handset ‡No Hands-free Mode				
Polycom	***SP IP 331***	0.92	−2.23	1.36	1.95
	SP IP 550	−2.71	−2.25	1.15	1.55
	SP IP 601	−0.68	−2.72	0.49	1.25
	SP IP 650	−2.69	−2.24	1.21	1.92
Windows	**Bria** (3.5.3.2)		−0.024		0.403
	Linphone (3.6.1)		−0.003		68.3
	X-Lite (3.0)		−0.065		0.393
Linux	**Ekiga** (3.3.2)		−130		1630
	SFLPhone (1.1.0)		0.004		0.045

models is the number of supported lines (i.e., 550 model supports 4 lines, whereas 650 model supports 6 lines).

7.2 Fingerprinting

The `REGISTER` refresh timer value is driven by a device's SIP configuration parameters as set by the service provider in its device configuration files. However, calling devices have its own unique way of maintaining the refresh `REGISTER` timer. This uniqueness is measured in terms of the device's clock skew. Figure 6 plots refresh offset data points of each of the 11 hardphone models and its corresponding clock skew. We observed that clock source of SIP timers are more precise in the case of Avaya and Polycom phones as compared to other phone vendors. It is possible that Avaya and Polycom phones timers are driven by a media clock, whereas all other hardphones are using a less precise system clock. We also observe that refresh `REGISTER` behavior of `Polycom SoundPoint IP 601` phone is unique among all other phones. To adjust the deviation of refresh timer value, the phone retransmits a duplicate (with same CSeq number) refresh `REGISTER` message from time-to-time. These duplicate retransmissions occur at $T1$ (i.e., 500 ms) or $2 * T1$ time interval from the previous `REGISTER` message. After retransmitting a duplicate `REGISTER` message, the phone resumes the same refresh timer value once again. Toward fingerprinting a softphone-based calling device, we install same X-Lite SIP client software on five different AC powered laptop computers. The two laptop computers (represented as A and B) are identical with respect to *Windows 8* OS and hardware specifications. Irrespective of the ethernet or wireless WiFi connection modes, refresh `REGISTER` messages can be used successfully to fingerprint a calling device as shown in Figure 7. A softphone registering from a laptop computer



(a.) Ethernet Connection (b.) WiFi Connection

Figure 7: Refresh REGISTER Offset Data Points (X-Lite)

shows two different clock skew values depending upon the connection mode (i.e., WiFi vs. Ethernet) used to access the Internet.

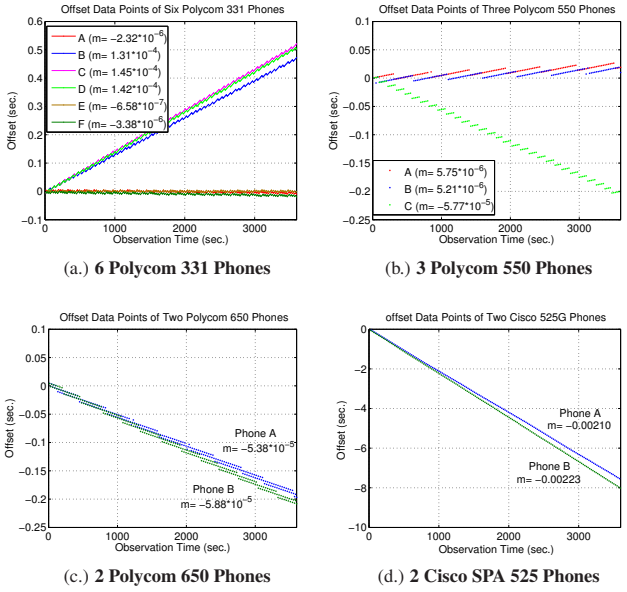
7.3 Further Experiments With Hardphones

One may question whether the previous analysis results will still hold true if we select several devices from the same class i.e., a set of phones from the same vendor and with same model number. In this set of experiments, we select a group of 6 `Polycom SoundPoint IP 331` and another group of 3 `Polycom Sound Point IP 550` phones. The silence RTP packets of all of the phones from both groups are analyzed, and their results are compared in Table 2. The experimental results demonstrate that analysis of acoustic features produce similar results across the same class of devices. As mentioned earlier, the hands-free mode on both of the phone models behave similarly; whereas, handset mode is quite different.

Table 2: Analysis Of Hardphones (Same Class Devices)

		DC Offset		Silence Energy	
Phone	Phone	handset	hands-free	handset	hands-free
Vendor	Model	$*10^{-4}$	$*10^{-4}$	$*10^{-7}$	$*10^{-7}$
Polycom	***331 (A)***	0.93	−2.23	1.35	1.92
	331 (B)	0.87	−2.23	1.27	1.63
	331 (C)	0.89	−2.24	1.37	1.85
	331 (D)	0.91	−2.24	1.39	1.70
	331 (E)	0.91	−2.23	1.45	1.86
	331 (F)	0.93	−2.22	1.40	1.67
Polycom	***550 (A)***	−2.73	−2.25	1.63	1.65
	550 (B)	−2.70	−2.24	1.42	1.78
	550 (C)	−2.73	−2.25	1.47	1.31

For fingerprinting, the clock skew measurement experiment of individual devices is performed under the following conditions: 1) Each group of devices share the same device configuration files; 2) All of the devices use NTP to synchronize their system clock; 3) The device access location remains fixed (i.e., phones register from the same location); 4) The signaling and media streams flow through the same SBC. For each individual phone, the offset data points collected from refresh `REGISTER` messages and the corresponding clock skew is plotted in Figure 8 (a.) and (b.), for `331` and `550` models, respectively. Although, each phone has its own unique value of clock skew; we can still observe how these clock skew values are divided into two groups. To find a plausible reason for this groupings of clock skew values, we notice that `Polycom 331` phones `A, E, F` have a manufacturing stamp of *1668-12379-001 Rev F2*, and `B, C, D` have a manufacturing stamp of *1668-12379-001 Rev D*. Similarly, `Polycom 550` phones `A, B` have



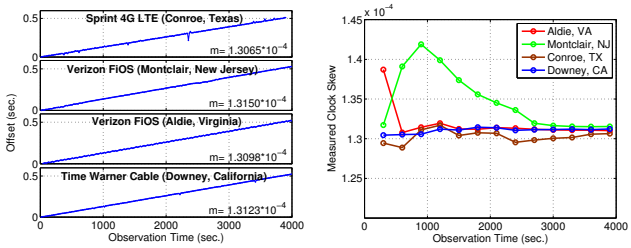
(a.) **6 Polycom 331 Phones** (b.) **3 Polycom 550 Phones**

(c.) **2 Polycom 650 Phones** (d.) **2 Cisco SPA 525 Phones**

Figure 8: Measuring Clock Skew - Homogeneous Devices

a manufacturing stamp of *1668-12540-001 Rev D*, and phone `C` has a manufacturing stamp of *1668-12540-001 Rev A*. To further explore this unique behavior, we selected two other sets of phones – two `Polycom SoundPoint IP 650` with the same manufacturing stamp *1668-12640-001 Rev E*, and two `Cisco SPA 525G` phones with version number (i.e., *VID V01*). Both Polycom and Cisco phones show similar clock skew values if the phones belong to same manufacturing batch as shown in Figure 8 (c.) and (d.), respectively. It is apparent that each phone's manufacturing release version represents some hardware changes affecting the clock precision and hence phone's refresh timer values.

Device Access Locations: Let us consider the behavior of a nomadic subscriber where the same physical calling device is connected to a VoIP service provider network (through the same access SBC located in Greenville, SC) from different locations using broadband Internet connection. For example, in our experiments, the `Polycom SoundPoint IP 331` is connected to service provider network from four US cities (e.g., Aldie, Virginia; Montclair, New Jersey; Conroe, Texas; Downey, California). It should be noted that the classification of remote calling device is based on payload analysis, and hence it remains independent of a device's access location. However, refresh `REGISTER` fingerprinting technique depends upon the packet's arrival time as recorded by the DI module, and hence depends upon the network path characteristics. For each access location, we plot offset data points and clock skew



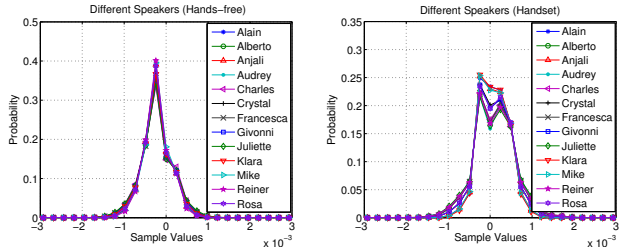
(a.) **Network Path Dependence** (b.) **Fingerprinting Sensitivity**

Figure 9: Registration From Different Access Locations

value in Figure 9 (a.). The measured value of the clock skew from four access locations demonstrate the applicability and robustness of a timer-based fingerprinting approach.

Fingerprinting Sensitivity: As an important attribute, we also study the sensitivity of device fingerprinting scheme i.e., how quickly can we fingerprint a target device. The offset data points of refresh `REGISTER` messages are analyzed every 300 seconds time interval (i.e., at $300, 600, 900, ..., 3600$ seconds) to estimate the clock skew. For each of the access locations, Figure 9 (b.) plots estimated clock skew values versus observation time. Within first 300 seconds of observation time, we could determine the device's manufacturing stamp. As time progresses, the clock skew values stabilize and become more accurate as shown in Figure 9 (b.). The accuracy of clock skew is highly dependent upon the device's access location (e.g., broadband access device, broadband access method, network path conditions etc.), and may require 15 to 45 minutes of observation time. However, it should be noted that the device's class and its manufacturing batch number determination is sufficient enough to weed out unauthorized calling devices within 5 minutes.

Subscriber's Calling Behavior: How does the proposed scheme perform when callers speak in different languages (such as English, Spanish, German, Italian, French), or in different accents (such as American, British or Indian English), or have different gender (i.e., male vs. female speakers)? We select 13 different telephone callers whose voices are synthetically created using *AT&T Natural Voices TTS System* [2]. The .wav files of these individual callers are



(a.) **Hands-free** (b.) **Handset**

Figure 10: Distribution Of Silence Samples

played in front of the same physical device `Polycom SoundPoint IP 331`. Each individual file (representing a caller) initiates a short call with a callee in similar fashion saying *"Hello! Hello! How are you? Is it a good time to talk with you?"* (either in English or

using their own native language). The initial 15 seconds of audio payload from each individual call is captured from the respective RTP streams and analyzed to identify the calling device.

Table 3: Studying Effect Of Caller's Calling Behavior

		DC Offset		Silence Energy	
Caller	Caller	handset	hands-free	handset	hands-free
Name (Sex)	Language	$*10^{-4}$	$*10^{-4}$	$*10^{-7}$	$*10^{-7}$
Crystal (Female)	***US English***	0.922	−2.228	1.53	1.65
Mike (Male)	***US English***	0.910	−2.223	1.43	1.70
Charles (Male)	***UK English***	0.931	−2.228	1.56	2.06
Audrey (Female)	***UK English***	0.952	−2.234	1.62	2.09
Anjali (Female)	***IND. English***	0.926	−2.224	1.39	1.76
Rosa (Female)	***Spanish***	0.929	−2.231	1.47	1.86
Alberto (Male)	***Spanish***	0.916	−2.223	1.62	2.06
Klara (Female)	***German***	0.940	−2.228	1.40	1.99
Reiner (Male)	***German***	0.943	−2.229	1.59	1.62
Francesca (Female)	***Italian***	0.961	−2.225	1.68	2.07
Giovanni (Male)	***Italian***	0.937	−2.228	1.55	2.08
Alain (Male)	***French***	0.959	−2.226	1.42	1.96
Juliette (Female)	***French***	0.967	−2.231	1.65	2.07

Our expectation is that because device identification relies on silence samples, we could still get the same device profile irrespective of the caller, and how (s)he speaks. Figure 10 shows silence samples distribution of all 13 different speakers in both hands-free and handset modes. For the same set of experiments, the DC Offset and Silence Energy of each individual callers are tabulated in Table 3. The test results demonstrate successful classification of the remote calling device, irrespective of subscriber's calling behavior.

7.4 Further Discussion And Future Work

Now we discuss if it is possible to circumvent the proposed device authorization scheme? Theoretically, yes, it is possible. However, in real-world scenarios, it is like looking for a needle in a haystack. To be successful, a remote hacker has to find a calling device that has: 1) same vendor and model number; 2) same refresh `REGISTER` timer value; 3) a very close value of clock skew (as measured by using refresh `REGISTER` messages). If all of the above conditions are met by a spurious device within a single trial, a remote hacker can originate fraudulent calls.

The acoustic features are studied within silence samples, does it mean the proposed scheme will not work under voice activity detection (VAD)? No, it is still valid under VAD. When the VAD detects a drop-off of speech amplitude, it waits a fixed amount of time before it stops putting speech frames in packets. This fixed amount of time is known as the *hangover* and is typically 200 ms [9]. Therefore, we still get silence samples for our analysis. However, we should note that instead of phones (i.e., access-side devices), VAD is commonly used at the core-side elements such as switches, gateways, call managers etc.

We currently analyze the most widely used `G.711` encoded audio streams only. The other less widely used codecs such as `G.722`, `G.729` will be considered as part of our future work. We are also interested in extending our analysis to include: 1) the effect of abnormal ambient environments where background noise level is much higher than in a normal office environment; 2) the registration behavior of a softclient where laptop connection to the Internet is switched back and forth between ethernet and wireless WiFi modes; 3) the registration behavior of a softclient when the laptop is running on battery power.

8. RELATED WORK

To date, most of the industry and academic efforts to address VoIP related attacks are focused on: 1) determining the identity and trust value of callers; 2) developing stronger authentication mechanisms; 3) analyzing the signaling messages to ascertain the true nature of call originating sources. Dantu et. al. [8] use the Bayesian algorithm to compute the reputation value of a caller based on his past behavior and callee's feedback. Rebahi et. al. [23] derive caller's reputation value by consulting SIP repositories along the call path from call's source to its destination. Wu et. al. [29] propose a spam detection approach involving user-feedback and semi-supervised clustering technique to differentiate between spam and legitimate calls. However, the derivation of caller's reputation value requires building a social network; the notion of user's feedback requires modification of SIP clients and an extension of SIP protocol [19]. Furthermore, these schemes rely on caller's identity which can be spoofed. Kayote Inc. [15] proposes a central *Trust Anchor* that is responsible for certifying and asserting relevant security information about the calling party. However, the central authority could become a single point of failure, an attractive and potential target for denial-of-service attacks, and a bottleneck for performance. Recently, Balasubramaniyan et. al. [4] propose a PinDr0p method to protect caller-ID based on call provenance. This method determines the traversal of a call through different service provider networks (i.e., VoIP, cellular, and PSTN). It is based on call audio features (such as applied voice codecs, packet loss and noise profile) bringing the networks information which it has traversed. Within IETF, a Secure Telephone Identity Revisited (STIR) group is formed to tackle problems related with the lack of security mechanisms for attesting the origins of real-time communications. The working group specifies a SIP header-based authorization mechanism to verify whether the originator of a SIP session is in fact authorized to use the claimed source telephone number [22]. However, our present work provides an alternative solution without overhauling the infrastructure, or asking for modification to the SIP protocol. As it relates to device fingerprinting, we are aware of Yan et. al.'s [30] SIP message format method and Kohno et. al.'s [16] timestamp-based remote physical device fingerprinting method. As discussed earlier, measuring the device's clock skew from its RTP timestamps has low accuracy and not suitable for device authorization scheme. Yan et al.'s fingerprinting scheme expects that a malicious software has different implementation than a legitimate client and consequently, the SIP message formats are structured differently. However, with some extra efforts, a hacker can construct a SIP message as it originates from a legitimate client.

9. CONCLUSION

This paper presents a remote calling device identification scheme encompassing both *classification* and *fingerprinting* techniques. By passive and remote observation of signaling and media streams, it is not only possible to determine the device's manufacturer name, model number, and even manufacturing batch number, but also to fingerprint a remote device with a high degree of accuracy. Our real-world experiments and encouraging results compellingly illustrate the possibility of a powerful notion of establishing a relationship between user ID (e.g., telephone number) and its authorized calling device(s). This aspect of *device authorization* in addition to the already existing *user authentication* is capable of preventing a plethora of VoIP attacks on VoIP service provider networks and their customer sites, and hence even more ominously suggests that there is a security solution such as device authorization that we have yet to integrate into our current VoIP security model.

10. REFERENCES

[1] R. Aaronoff. Communications Fraud Control Association (CFCA) Annonunces Results of Worldwide Telecom Fraud Survey. http://www.cfca.org/pdf/survey/Global%20Fraud_Loss_Survey2011.pdf, 2013.

[2] AT&T Labs. AT&T Natural Voices Text-to-Speech Demo. http://www2.research.att.com/ ttsweb/tts/demo.php, 2013.

[3] C. Avants. ACME Packet's New Toll-Fraud Mitigation Tool. http://chrisavants.com/acme-packets-new-toll-fraud-mitigation-tool/, 2013.

[4] V. A. Balasubramaniyan, A. Poonawalla, M. Ahamad, M. T. Hunter, and P. Traynor. Pindr0p: Using single-ended audio features to determine call provenance. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, 2010.

[5] A. Boone. Toll fraud is alive and well. http://www.networkworld.com/news/tech/2009/092909-tech-update.html, 2009.

[6] M. Collier. Mark Collier's VoIPUC Security Blog. http://voipsecurityblog.typepad.com/marks_voip_security_blog/, 2013.

[7] A. Dainotti, A. King, K. Claffy, F. Papale, and A. Pescapè. Analysis of a "/0" Stealth Scan from a Botnet. In *Internet Measurement Conference (IMC)*, pages 1–14, Nov 2012.

[8] R. Dantu and P. Kolan. Detecting spam in voip networks. In *Proceedings of the Steps to Reducing Unwanted Traffic on the Internet on Steps to Reducing Unwanted Traffic on the Internet Workshop*, 2005.

[9] J. Davidson, J. F. Peters, M. Bhatia, S. Kalidindi, and S. Mukherjee. *Voice over IP Fundamentals*. Cisco Press, 2nd edition, 2006.

[10] T. Davies. How to make your VoIP secure #fraud. http://www.trefor.net/2013/01/24/how-to-make-your-voip-secure/, 2013.

[11] S. Dawson. What is Dither? . http://www.hifi-writer.com/he/dvdaudio/dither.htm, 2013.

[12] B. Huston. Just a Reminder, SIP is a Popular Scanning Target. http://stateofsecurity.com/?p=3171, 2013.

[13] ITU-T. Pulse Code Modulation (PCM) of Voice Frequencies. http://www.itu.int/rec/T-REC-G.711-198811-I/en, 2013.

[14] jpcap. Network Packet Capture Facility for Java. http://sourceforge.net/projects/jpcap/, 2013.

[15] Kayote Networks. The Threat of SPIT. http://www.kayote.com/, 2007.

[16] T. Kohno, A. Broido, and K. C. Claffy. Remote Physical Device Fingerprinting. In *IEEE Symposium on Security and Privacy*, May 2005.

[17] D. Lanman. Sound Level Meter. http://www.mathworks.com/matlabcentral/fileexchange/9603-sound-level-meter, 2013.

[18] P. McNeil. Announcing sipShield, the security plugin for SBCs! https://community.acmepacket.com/t5/Security-Expert-Days/Announcing-sipShield-the-security-plugin-for-SBCs/m-p/7340, 2013.

[19] S. Niccolini, S. Tartarelli, M. Stiemerling, and S. Srivastava. SIP Extensions for SPIT identification. , IETF Network Working Group, Work in Progress, 2007.

[20] S. Orman. Hackers Hit DeSoto Phone System, Run Up $23K Bill. http://www.localmemphis.com/news/local/story/Hackers-Hit-DeSoto-Phone-System-Run-Up-23K-Bill/t_qvwmGL8kiwRQG1fDFkXg.cspx, 2013.

[21] peers. Peers Java SIP Softphone. http://peers.sourceforge.net/, 2013.

[22] J. Peterson, H. Schulzrinne, and H. Tschofenig. Secure Origin Identification: Problem Statement, Requirements, and Roadmap. , IETF Network Working Group, Work in Progress, 2013.

[23] Y. Rebahi and A. Al-Hezmi. Spam Prevention for Voice over IP. http://colleges.ksu.edu.sa/ComputerSciences/Documents/NITS/ID143.pdf, 2007.

[24] M. Rockwell. Sen. Schumer: Al Qaeda-linked phone hackers costing NY small businesses. http://www.gsnmagazine.com/node/28198?c=communications, 2013.

[25] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261, IETF Network Working Group, 2002.

[26] L. Serrano. The newest SIP & PROVIDERS SCANNER !!! . http://www.youtube.com/watch?v=GrxDXmvh13c, 2013.

[27] M. Stocco. The Less Than Friendly-scanner, Sipvicious. http://advantia.ca/weblog/less-than-friendly-scanner-sipvicious, 2013.

[28] thejournal.ie. ComReg warns businesses of increase in phone hacking. http://www.thejournal.ie/comreg-warns-businesses-of-increase-in-phone-hacking-723460-Dec2012/, 2012.

[29] Y.-S. Wu, S. Bagchi, N. Singh, and R. Wita. Spam Detection in Voice-Over-IP Calls through Semi-Supervised Clustering. In *IEEE Dependable Systems and Networks Conference (DSN 2009)*, June-July 2009.

[30] H. Yan, K. Sripanidkulchai, H. Zhang, Z.-Y. Shae, and D. Saha. Incorporating Active Fingerprinting into SPIT Prevention Systems. In *3rd Workshop on Securing Voice over IP*, June 2006.

APPENDIX

A. VOIP DEVICE MANAGEMENT

At the customer (i.e., telephone subscriber) site there are many types of access devices such as softphones, hardphones, and integrated access devices (IADs) etc. that need configuration profiles, firmware, and other files to provide proper operations of call services. To ease the deployment, provisioning, and management of customer end devices, the VoIP service providers control these access devices centrally from their networks. For example, we take a real world example of a phone vendor *Polycom* and a softswitch vendor *Broadsoft*[6].

As shown in Figure 11, a user telephone number 7775551234 is provisioned within the softswitch system. This telephone number is associated with a specific device profile named `Polycom Soundpoint IP 650` (a Polycom phone model). For this particular user, the phone specific configuration files are created and stored in the profile server (a type of Broadsoft server used to store device configuration files of telephone subscribers). When a phone reboots, it authenticates itself to the Broadsoft server and fetches (using ftp, http, or https) corresponding configuration files. The Figure 12 shows a screenshot of supported Polycom SoundPoint IP phones by the Broadsoft softswitch. Similarly, we can find a

[6]BroadSoft is deployed in more than 450 telecommunications service providers' networks and serves 15 of the top 25 largest telecommunications carriers.

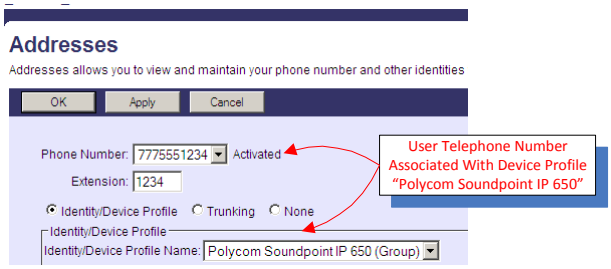


Figure 11: Screenshot (From Broadsoft Softswitch) Showing Relationship Between Subscriber And Its Device Type

list of hundreds of devices from various other vendors whose interoperability has been tested against the softswitch and published as supported devices.

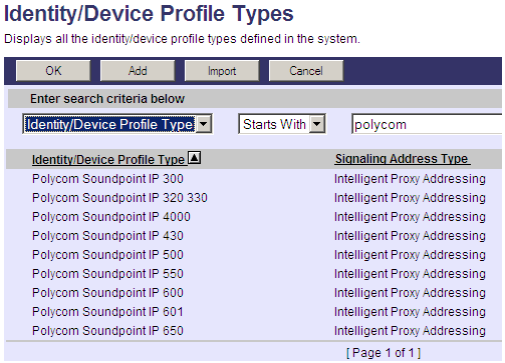


Figure 12: Screenshot Showing Various Polycom Phones Supported By The Broadsoft Softswitch

Now we discuss how this relationship between user and device type is established. When a user account (i.e., a telephone number) is provisioned within the system, at that point the service provider already knows the device to be associated with the telephone number. This relationship exists to ease the device management from the service provider perspective and be able to offer or restrict advanced call features that can be enabled on that device depending upon user's subscribed features (such as shared call appearance, simring, huntgroup etc.).

B. VOIP ARCHITECTURE

In today's IP telephony world, the VoIP service providers operate in partially closed environments and are connected to each other through the public switched telephone network (PSTN) as shown in Figure 13. In a partial closed environment, the SIP proxy server resources are accessed by its own authenticated subscribers only. The authentication of call requests is possible because user accounts (containing authentication credentials, subscribed call features, and policy etc.) are stored locally. However, VoIP service providers are pushing hard to opt for open architecture of VoIP service where service providers can interact with each other through the IP-based peering points. It provides an ability for individual subscribers to connect with each other without traversing the PSTN cloud.

C. RTP-BASED DEVICE FINGERPRINTING

To study the network path dependence on the voice stream, the same physical calling device (`Polycom SoundPoint IP 331` Phone) is used to make calls from several US cities using Verizon FiOS,

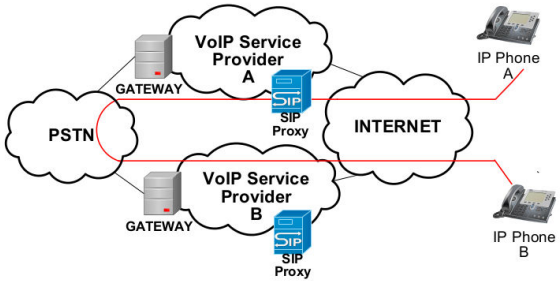


Figure 13: VoIP Service Provider Network

Time Warner Cable, Sprint 4G LTE, and campus network connections. This is a realistic calling behavior of a nomadic subscriber, and also the way most common fraud attacks are launched. The nomadic telephony service allows subscribers to move their VoIP phones from one location to another with the access of high-speed Internet connection. At service provider's SBC (located in Greenville, South Carolina), offset data points are derived from the RTP stream. Each of the access locations introduces an inherent noise in offset data points differently and independently of each other as shown in Figure 14.

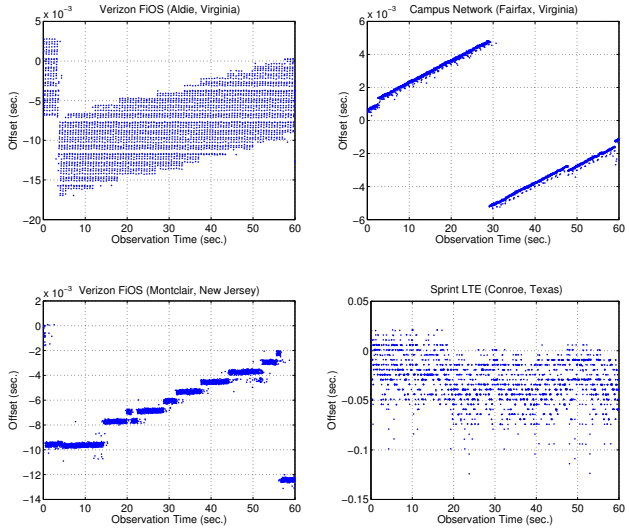


Figure 14: Offset Data Points From `RTP` Timestamp (Same Physical Device `Polycom SoundPoint IP 331` Calling From Different Locations)