AuthLoop: End-to-End Cryptographic Authentication for Telephony over Voice Channels

Bradley Reaves, Logan Blue, and Patrick Traynor, *University of Florida*

https://www.usenix.org/conference/usenixsecurity16/technical-sessions/presentation/reaves

This paper is included in the Proceedings of the
25th USENIX Security Symposium
August 10–12, 2016 • Austin, TX
ISBN 978-1-931971-32-4

AuthLoop: Practical End-to-End Cryptographic Authentication for Telephony over Voice Channels

Bradley Reaves
University of Florida
reaves@ufl.edu

Logan Blue
University of Florida
bluel@ufl.edu

Patrick Traynor
University of Florida
traynor@cise.ufl.edu

Abstract

Telephones remain a trusted platform for conducting some of our most sensitive exchanges. From banking to taxes, wide swathes of industry and government rely on telephony as a secure fall-back when attempting to confirm the veracity of a transaction. In spite of this, authentication is poorly managed between these systems, and in the general case it is impossible to be certain of the identity (i.e., Caller ID) of the entity at the other end of a call. We address this problem with AuthLoop, the first system to provide cryptographic authentication solely within the voice channel. We design, implement and characterize the performance of an in-band modem for executing a TLS-inspired authentication protocol, and demonstrate its abilities to ensure that the explicit single-sided authentication procedures pervading the web are also possible on all phones. We show experimentally that this protocol can be executed with minimal computational overhead and only a few seconds of user time (≈ 9 instead of ≈ 97 seconds for a naïve implementation of TLS 1.2) over heterogeneous networks. In so doing, we demonstrate that strong end-to-end validation of Caller ID is indeed practical for all telephony networks.

1 Introduction

Modern telephony systems include a wide array of end-user devices. From traditional rotary PSTN phones to modern cellular and VoIP capable systems, these devices remain the de facto trusted platform for conducting many of our most sensitive operations. Even more critically, these systems offer the sole reliable connection for the majority of people in the world today.

Such trust is not necessarily well placed. Caller ID is known to be a poor authenticator [59, 18, 67], and yet is successfully exploited to enable over US$2 Billion in fraud every year [28]. Many scammers simply block their phone number and exploit trusting users by asserting an identity (e.g., a bank, law enforcement, etc.), taking advantage of a lack of reliable cues and mechanisms to dispute such claims. Addressing these problems will require the application of lessons from a related space. The Web experienced very similar problems in the 1990s, and developed and deployed the Transport Layer Security (TLS) protocol suite and necessary support infrastructure to assist with the integration of more verifiable identity in communications. While by no means perfect and still an area of active research, this infrastructure helps to make a huge range of attacks substantially more difficult. Unfortunately, the lack of similarly strong mechanisms in telephony means that *not even trained security experts can currently reason about the identity of other callers.*

In this paper, we address this problem with AuthLoop.[1] AuthLoop provides a strong cryptographic authentication protocol inspired by TLS 1.2. However, unlike other related solutions that assume Internet access (e.g., Silent Circle, RedPhone, etc [24, 73, 25, 5, 3, 6, 1, 74, 7]), accessibility to a secondary and concurrent data channel is not a guarantee in many locations (e.g., high density cities, rural areas) nor for all devices, mandating that a solution to this problem be network agnostic. Accordingly, AuthLoop is designed for and transmitted over the only channel certain to be available to all phone systems — audio. The advantage to this approach is that it requires no changes to any network core, which would likely see limited adoption at best. Through the use of AuthLoop, users can quickly and strongly identify callers who may fraudulently be claiming to be organizations including their financial institutions and their government [28].

We make the following contributions:

[1] A name reminiscent of the "Local Loop" used to tie traditional phone systems into the larger network, we seek to tie modern telephony systems into the global authentication infrastructure that has dramatically improved transaction security over the web during the past two decades.

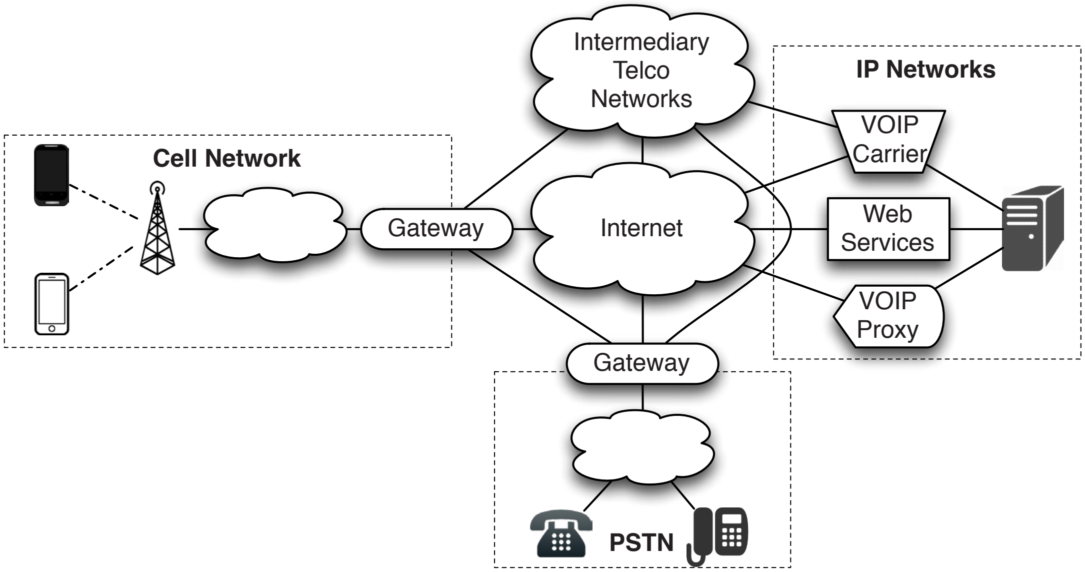


Figure 1: A high-level representation of modern telephony systems. In addition to voice being transcoded at each gateway, all identity mechanisms become asserted rather than attested as calls cross network borders. A strong end-to-end authentication must be designed aware of all such limitations.

- **Design a Complete Transmission Layer:** We design the first codec-agnostic modem that allows for the transmission of data across audio channels. We then create a supporting link layer protocol to enable the reliable delivery of data across the heterogeneous landscape of telephony networks.
- **Design AuthLoop Authentication Protocol:** After characterizing the bandwidth limitations of our data channel, we specify our security goals and design the AuthLoop protocol to provide explicit authentication of one party (i.e., the "Prover") and optionally weak authentication of the second party (i.e., the "Verifier").
- **Evaluate Performance of a Reference Implementation:** We implement AuthLoop and test it using three representative codecs — G.711 (for PSTN networks), AMR (for cellular networks) and Speex (for VoIP networks). We demonstrate the ability to create a data channel with a goodput of 500 bps and bit error rates averaging below 0.5%. We then demonstrate that AuthLoop can be run over this channel in an average of 9 seconds (which can be played below speaker audio), compared to running a direct port of TLS 1.2 in an average of 97 seconds (a 90% reduction in running time).

The remainder of this paper is organized as follows: Section 2 provides background information and related work; Section 3 presents the details of our system including lower-layer considerations; Section 4 discusses our security model; Section 5 formally defines the AuthLoop protocol and parameterizes our system based on the modem; Section 6 discusses our prototype and experimental results; Section 7 provides additional discussion about our system; and Section 8 provides concluding remarks.

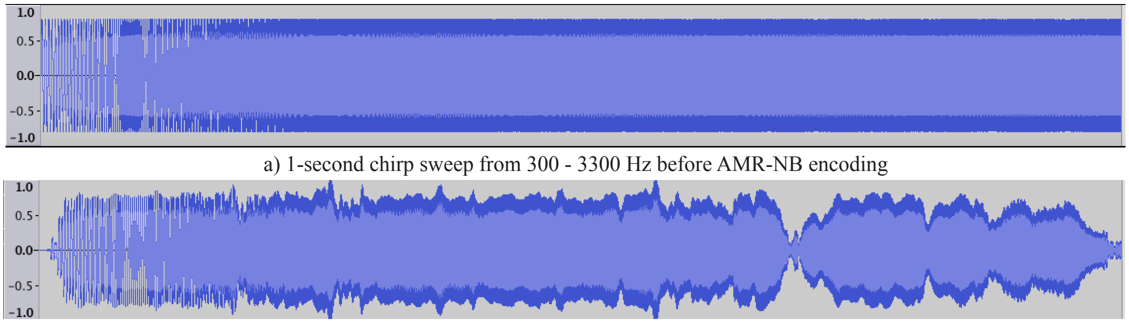
2 Background and Related Work

In this section, we provide an overview of modern telephony networks and review current and proposed practices of authentication in those networks.

2.1 Modern Telephony Networks

The landscape of modern telephony is complex and heterogeneous. Subscribers can receive service from mobile, PSTN and VoIP networks, and calls to those subscribers may similarly originate from networks implementing any of the above technologies. Figure 1 provides a high-level overview of this ecosystem.

While performing similar high-level functionality (i.e., enabling voice calls), each of these networks is built on a range of often incompatible technologies. From circuit-switched intelligent network cores to packet switching over the public Internet, very little information beyond the voice signal actually propagates across the borders of these systems. In fact, because many of these



a) 1-second chirp sweep from 300 - 3300 Hz before AMR-NB encoding

b) 1-second chirp sweep from 300 - 3300 Hz after AMR-NB encoding

Figure 2: A comparison of a signal (a) before and (b) after being encoded with the AMR codec. Note that while the entirety of the signal is within the range of allowable frequencies for call audio, the received signal differs significantly from its original form. It is therefore critical that a high-fidelity mechanism for delivering data over a mobile audio channel be designed.

networks rely on different codecs for encoding voice, one of the major duties of gateways between these systems is the transcoding of audio. Accordingly, voice encoded at one end of a phone call is unlikely to have the same (or even similar) bitwise representation when it arrives at the client side of the call. As evidence, the top plot of Figure 2 shows a sweep of an audio signal from 300 to 3300 Hz (all within the acceptable band) across 1 second. The bottom plot shows the same signal after is has been encoded using the Adaptive Multi-Rate (AMR) audio codec used in cellular networks, resulting in a dramatically different message. This massive difference is a result of the voice-optimized audio codecs used in different telephony networks. Accordingly, successfully performing end-to-end authentication will require careful design for this non-traditional data channel.

One of the few pieces of digital information that can be optionally passed between networks is the Caller ID. Unfortunately, the security value of this metadata is minimal — such information is asserted by the source device or network, but never validated by the terminating or intermediary networks. As such, an adversary is able to claim any phone number (and therefore identity) as its own with ease. This process requires little technical sophistication, can be achieved with the assistance of a wide range of software and services, and is the enabler of greater than US$2 Billion in fraud annually [28].

2.2 Authentication in Telephony Networks

Authentication has been the chief security concern of phone networks since their inception because of its strong ties to billing [69]. Little effort was taken for authentication in traditional landline networks as detecting billable activity on a physical link limited the scalability of attacks. First generation (1G) cellular systems were the first to consider such mechanisms given the multi-user nature of wireless spectrum. Unfortunately, 1G authentication relied solely on the plaintext assertion of each user's identity and was therefore subject to significant fraud [53]. Second generation (2G) networks (e.g., GSM) designed cryptographic mechanisms for authenticating users to the network. These protocols failed to authenticate the network to the user and lead to a range of attacks against subscribers [44, 26, 19, 68]. Third and fourth generation (3G and 4G) systems correctly implement mutual authentication between the users and providers [11, 12, 13]. Unfortunately, all such mechanisms are designed to allow accurate billing, and do little to help users identify other callers.

While a number of seemingly-cellular mechanisms have emerged to provide authentication between end users (e.g., Zphone, RedPhone) [24, 73, 25, 5, 3, 6, 1, 74, 7, 31, 30] , these systems ultimately rely on a data/Internet connection to work, and are themselves vulnerable to a number of attacks [63, 52]. Accordingly, there remains no end-to-end solution for authentication *across voice* networks (i.e., authentication with any non-VoIP phone is not possible).

Mechanisms to deal with such attacks have had limited success. Websites have emerged with reputation data for unknown callers [2]; however, these sites offer no protection against Caller-ID spoofing, and users generally access such information after such a call has occurred. Others have designed heuristic approaches around black lists [4], speaker recognition [71, 16, 72, 17, 66, 41], channel characterization [18, 54], post hoc call data records [58, 47, 23, 40] and timing [61]. Unfortunately, the fuzzy nature of these mechanisms may cause them to fail under a range of common conditions including congestion and evasion.

Authentication between entities on the Internet generally relies on the use of strong cryptographic mechanisms. The SSL/TLS suite of protocols are by far the most widely used, and help provide attestable identity for applications as diverse as web browsing, email, instant messaging and more. SSL/TLS are not without their own issues, including a range of vulnerabilities across different versions and implementations of the protocols [48, 27, 75, 34], weaknesses in the model and deployment of Certificate Authorities [57, 36, 37, 38, 29, 39, 20], and usability [55, 32, 60, 35, 65, 14, 15]. Regardless of these challenges, these mechanisms provide more robust means to reason about identity than the approaches used in telephony.

Telephony can build on the success of SSL/TLS. However, these mechanisms can not simply be built on top of current telephony systems. Instead, and as we will demonstrate, codec-aware protocols that are optimized for the limited bitrate and higher loss of telephony systems must be designed.

3 Voice Channel Data Transmission

To provide end-to-end authentication across any telephone networks, we need a way to transfer data over the voice channel. The following sections detail the challenges that must be addressed, how we implemented a modem that provides a base data rate of 500bps, and how we developed a link layer to address channel errors. We conclude with a discussion of what these technical limitations imply for using standard authentication technologies over voice networks.

3.1 Challenges to Data Transmission

Many readers may fondly remember dial-up Internet access and a time when data transmission over voice channels was a common occurrence. In the heyday of telephone modems, though, most voice channels were connected over high-fidelity analog twisted pair. Although the voice channel was band limited and digital trunks used a low sample rate of 8kHz, the channel was quite “well behaved” from a digital communications and signal processing perspective.

In the last two decades, telephony has been transformed. Cellular voice and Internet telephony now comprise a majority of all voice communications; they are not just ubiquitous, they are unavoidable. While beneficial from a number of perspectives, one of the drawbacks is that both of these modalities rely on heavily compressed audio transmission to save bandwidth. These compression algorithms – audio codecs – are technological feats, as they have permitted cheap, acceptable quality phone calls, especially given that they were developed during eras when computation was expensive. To do this, codec designers employed a number of technical and psychoacoustic tricks to produce acceptable audio to a human ear, and these tricks resulted in a channel poorly suited for (if not hostile to) the transmission of digital data. As a result, existing voice modems are completely unsuited for data transmission in cellular or VoIP networks.

Voice codecs present several challenges to a general-purpose modem. First, amplitudes are not well preserved by voice codecs. This discounts many common modulation schemes, including ASK, QAM, TCM, and PCM. Second, phase discontinuities are rare in speech, and are not effective to transmit data through popular voice codecs. This discounts PSK, QPSK, and other modulation schemes that rely on correct phase information. Furthermore, many codecs lose phase information on encoding/decoding audio, preventing the use of efficient demodulators that require correct phase (i.e., coherent demodulators). Because of the problems with amplitude and phase modulation, frequency-shift modulation is the most effective technique for transmitting data through voice codecs. Even so, many codecs fail to accurately reproduce input frequencies — even those well within telephone voicebands (300–3400 Hz). Our physical layer protocol addresses these challenges.

3.2 Modem design

The AuthLoop modem has three goals:

1. Support highest bitrate possible
2. At the lowest error rate possible
3. In the presence of deforming codecs

We are not the first to address transmission of data over lossy compressed voice channels. Most prior efforts [70, 51, 42] have focused on transmission over a single codec, though one project, Hermes [33] was designed to support multiple cellular codecs. Unfortunately, that project only dealt with the modulation scheme, and did not address system-level issues like receiver synchronization. Furthermore, the published code did not have a complete demodulator, and our own implementation failed to replicate their results. Thus, we took Hermes as a starting point to produce our modem.

Most modems are designed around the concept of modulating one or more parameters — amplitude, frequency, and/or phase — of one or more sine waves. Our modem modulates a single sine wave using one of three discrete frequencies (i.e. it is a frequency shift key, or FSK, modem). The selection of these frequencies is a key design consideration, and our design was affected by three design criteria.

First, our modem is designed for phone systems, so our choice of frequencies are limited to the 300–3400Hz

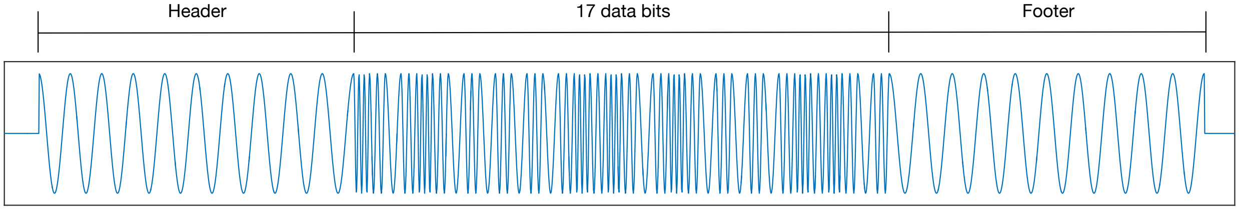


Figure 3: This 74ms modem transmission of a single frame demonstrates how data is modulated and wrapped in headers and footers for synchronization.

range because most landline and cellular phones are limited to those frequencies. Second, because we cannot accurately recover phase information for demodulation, our demodulation must be decoherent; the consequence is that our chosen frequencies must be separated by at least the symbol transmission rate [64]. Third, each frequency must be an integer multiple of the symbol frequency. This ensures that each symbol completes a full cycle, and it also ensures that each cycle begins and ends on a symbol boundary. This produces a continuous phase modulation, and it is critical because some voice codecs will produce artifacts or aliased frequencies in the presence of phase discontinuities. These constraints led to the selection of a 3-FSK system transmitting symbols at 1000 Hz using frequencies 1000, 2000, and 3000 Hz.

Unfortunately, 3-FSK will still fail to perform in many compressed channels simply because those channels distort frequencies, especially frequencies that change rapidly. To mitigate issues with FSK, we use a differential modulation: bits are encoded not as individual symbols, but by the relative difference between two consecutive symbols. For example, a “1” is represented by an increase in two consecutive frequencies, while a “0” is represented by a frequency decrease. Because we only have 3 frequencies available, we have to limit the number of possible consecutive increases or decreases to 2. Manchester encoding, where each bit is expanded into two “half-bits” (e.g. a “1” is represented by “10”, and “0” represented by “01”) limits the consecutive increases or decreases within the limit.

While these details cover the transmission of data, there are a few practical concerns that must be dealt with. Many audio codecs truncate the first few milliseconds of audio. In speech this is unnoticeable, and simplifies the encoding. However, if the truncated audio carries data, several bits will be lost every transmission. This effect is compounded if voice activity detection (VAD) is used (as is typical in VoIP and cellular networks). VAD distinguishes between audio and silence, and when no audio is recorded in a call VAD indicates that no data should be sent, saving bandwidth. However, VAD adds an additional delay before voice is transmitted again.

To deal with early voice clipping by codecs and VAD, we add a 20 ms header and footer at the end of each packet. This header is a 500 Hz sine wave; this frequency is orthogonal to the other 3 transmission frequencies, and is half the symbol rate, meaning it can be used to synchronize the receiver before data arrives. A full modem transmission containing 17 bits of random data can be seen in Figure 3.

To demodulate data, we must first detect that data is being transmitted. We distinguish silence and a transmission by computing the energy of the incoming signal using a short sliding window (i.e, the short-time energy). Then we locate the header and footer of a message to locate the beginning and end of a data transmission. Finally, we compute the average instantaneous frequency for each half-bit and compute differences between each bit. An increase in frequency indicates 1, a decrease indicates 0.

3.3 Link Layer

Despite a carefully designed modem, reception errors will still occur. These are artifacts created by line noise, the channel codec, or an underlying channel loss (e.g., a lost IP packet). To address these issues, we developed a link layer to ensure reliable transmission of handshake messages. This link layer manages error detection, error correction, frame acknowledgment, retransmission, and reassembly of fragmented messages.

Because error rates can sometimes be as high as several percent, a robust retransmission scheme is needed. However, because our available modem data rate is so low, overhead must be kept to a minimum. This rules out most standard transmission schemes that rely on explicit sequence numbers. Instead, our data link layer chunks transmitted frames into small individual blocks that may be checked and retransmitted if lost. We are unaware of other link layers that use this approach. The remainder of this subsection motivates and describes this scheme.

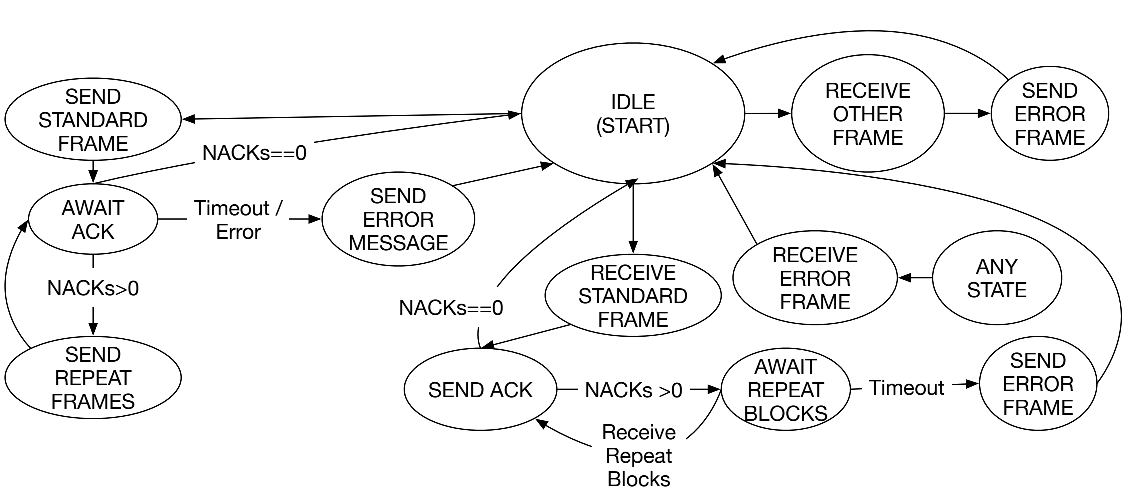


Figure 4: Link Layer State Machine

3.4 Framing and error detection

Most link layers are designed to transmit large (up to 12,144 bits for Ethernet) frames, and these channels either use large (e.g., 32-bit) CRCs[2] for error detection to retransmit the entire frame, or use expensive but necessary error correcting schemes in lossy media like radio. Error correcting codes recover damaged data by transmitting highly redundant data, often inflating the data transmitted by 100% or more. The alternative, sending large frames with a single CRC, was unlikely to succeed. To see why, note that:

$$P(CorrectCRC) = (1 - P(biterror))^{CRClength} \tag{1}$$

For a 3% bit error rate, the probability of just the CRC being undamaged is less than 38% — meaning two thirds of packets will be dropped for having a bad CRC independent of other errors. Even at lower loss rates, retransmitting whole frames for a single error would cause a massive overhead.

Instead, we divide each frame into 32-bit "blocks". Each block carries 29 bits of data and a 3-bit CRC. This allows short sections of data to be checked for errors individually and retransmitted, which is closer to optimal transmission. Block and CRC selection was not arbitrary, but rather the result of careful modeling and analysis. In particular, we aimed to find an optimal tradeoff between overhead (i.e., CRC length) and error detection. Intuitively, longer CRCs provide better error detection and reduce the probability of an undetected error. More formally, a CRC of length l can guarantee detection of up to HD bit errors[3] in a B-length block of data, and can detect more than HD errors probabilistically [43].

The tradeoff is maximizing the block size and minimizing the CRC length while minimizing the probability of a loss in the frame or the probability of an undetected error, represented by the following equations:

$$\begin{aligned} Pr(lostframe) &= 1 - Pr(successfulframe) \quad (2)\\ &= 1 - (1-p)^B \quad (3)\end{aligned}$$

$$Pr\begin{pmatrix} undetected \\ error \end{pmatrix} = 1 - \sum_{i=0}^{HD} \binom{B}{i} p^i (1-p)^{B-i} \tag{4}$$

where p represents the probability of a single bit error. The probability of undetected error is derived from the cumulative binomial distribution. Using these equations and the common bit error rate of 0.3% (measured in Section 6), we selected 32-bit blocks with a 3-bit CRC. We chose the optimal 3-bit CRC polynomial according to Koopman and Chakravarty [43]. These parameters give a likelihood of undetected error of roughly 0.013% , which will rarely affect a regular user. Even a call center user would see a protocol failure due to bit error once every two weeks, assuming 100 calls per day.

3.5 Acknowledgment and Retransmission

Error detection is only the first step of the error recovery process, which is reflected as a state machine in Figure 4.

When a message frame is received, the receiver computes which blocks have an error and sends an acknowledgment frame ("ACK") to the transmitter. The ACK frame contains a single bit for each block transmitted to indicate if the block was received successfully or not.

[2] A Cyclic Redundancy Check (CRC) is a common checksum that is formed by representing the data as a polynomial and computing the remainder of polynomial division. The polynomial divisor is a design parameter that must be chosen carefully.

[3] The Hamming distance of the transmitted and received data

Blocks that were negatively acknowledged are retransmitted; the retransmission will also be acknowledged by the receiver. This process will continue until all original blocks are received successfully.

By using a single bit of acknowledgment for each block we save the overhead of using sequence numbers. However, even a single bit error in an ACK will completely desynchronize the reassembly of correctly received data. Having meta-ACK and ACK retransmission frames would be unwieldy and inelegant. Instead, we transmit redundant ACK data as a form of error correction; we send ACK data 3 times in a single frame and take the majority of any bits that conflict. The likelihood of a damaged ACK is then:

$$\text{Block Count} \times 3 \times Pr(biterr)^2 \quad (5)$$

instead of

$$1-(1-Pr(biterr))^{\text{Block Count}} \quad (6)$$

Note that there are effectively distinct types of frames – original data, ACK data, retransmission data, and error frames. We use a four-bit header to distinguish these frames; like ACK data, we send three copies of the header to ensure accurate recovery. We will explore more robust error correcting codes in future work.

3.6 Naïve TLS over Voice Channels

With a modem and link layer design established, we can now examine how a standard authentication scheme — TLS 1.2 — would fare over a voice channel.

Table 1 shows the amount of data in the TLS handshakes of four popular Internet services: Facebook, Google, Bank of America, and Yahoo. These handshakes require from 41,000 to almost 58,000 bits to transmit, and this excludes application data and overhead from the TCP/IP and link layers. At 500 bits per second (the nominal speed of our modem), these transfers would require 83–116 seconds *as a lower bound*. From a usability standpoint, standard TLS handshakes are simply not practical for voice channels. Accordingly, a more efficient authentication protocol is necessary.

Table 1: TLS Handshake Sizes

Site Name	Total Bits	Transmission Time (seconds at 500bps)
Facebook	41 544	83.088
Google	42 856	85.712
Bank of America	53 144	106.288
Yahoo	57 920	115.840
Average	48 688	97.732

4 Security Model

Having demonstrated that data communication is possible but extremely limited via voice channels, we now turn our attention to defining a security model. The combination of our modem and this model can then be used to carefully design the AuthLoop protocol.

The goal of AuthLoop is to mitigate the most common enabler of phone fraud: claiming a false identity via Caller ID spoofing. This attack generally takes the form of the adversary calling the victim user and extracting sensitive information via social engineering. The attack could also be conducted by sending the victim a malicious phone number to call (e.g., via a spam text or email). An adversary may also attempt to perform a man in the middle attack, calling both the victim user and a legitimate institution and then hanging up the call on either when they wish to impersonate that participant. Finally, an adversary may attempt to perform a call forwarding attack, ensuring that correctly dialed numbers are redirected (undetected to the caller) to a malicious endpoint.

We base our design on the following assumptions. An adversary is able to originate phone calls from any telephony device (i.e., cellular, PSTN, or VoIP) and spoof their Caller ID information to mimic any phone number of their choosing. Targeted devices will either display this spoofed number or, if they contain a directory (e.g., contact database on a mobile phone), a name associated or registered with that number (e.g., “Bank of America”). The adversary can play arbitrary sounds over the audio channel, and may deliver either an automated message or interact directly with the targeted user. Lastly, the adversary may use advanced telephony features such as three-way calling to connect and disconnect parties arbitrarily. This model describes the majority of adversaries committing Caller ID fraud at the time of this work.

Our scenario contains two classes of participants, a Verifier (i.e., the user) and Prover (i.e., either the attacker of the legitimate identity owner). The adversary is active and will attempt to assert an arbitrary identity. As is common on the Web, we assume that Provers have certificates issued by their service provider[4] containing their public key and that Verifiers may have weak credentials (e.g., account numbers, PINs, etc) but do not have certificates. We seek to achieve the following security goals in the presence of this adversary:

1. **(G1) Authentication of Prover:** The Verifier should be able to explicitly determine the validity of an asserted Caller ID and the identity of the Prover without access to a secondary data channel.

[4] See Section 7 for details.

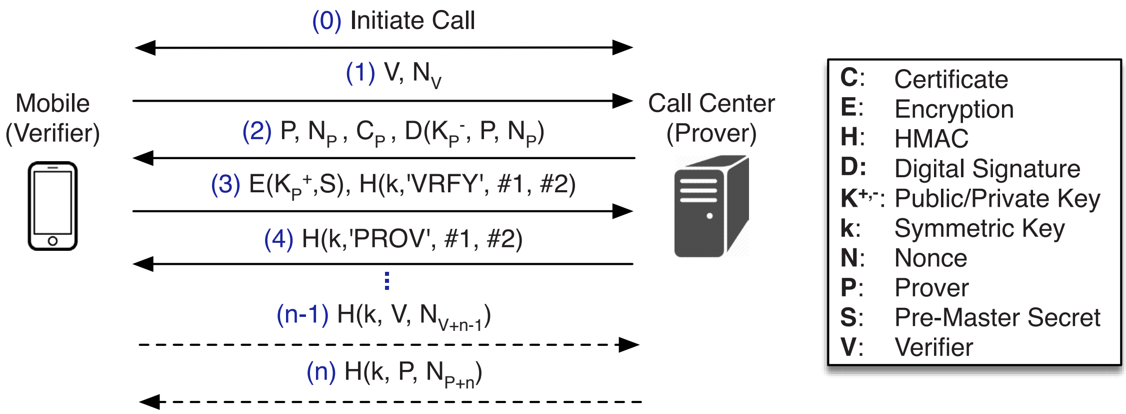


Figure 5: The AuthLoop authentication protocol. Solid arrows indicate the initial handshake message flows, and dotted arrows indicate subsequent authenticated "keep alive" messages. Note that #1 and #2 in messages 2 and 3 indicate that that contents of messages 1 and 2 are included in the calculation of the HMAC, as is done in TLS 1.2.

2. **(G2) Proof of Liveness:** The Prover and Verifier will be asked to demonstrate that they remain on the call throughout its duration.

Note that we do not aim to achieve voice confidentiality. As discussed in Section 2, the path between two telephony participants is likely to include a range of codec transformations, making the bitwise representation of voice vary significantly between source and destination. Accordingly, end-to-end encryption of voice content is not currently possible given the relatively low channel bitrate and large impact of transcoding. Solutions such as Silent Circle [7] and RedPhone [1] are able to achieve this guarantee strictly because they are VoIP clients that traverse *only* data networks and therefore do not experience transcoding. However, as we discuss in Section 7, our techniques enable the creation of a low-bandwidth channel that can be used to protect the confidentiality and integrity of weak client authentication credentials.

5 AuthLoop Protocol

This section describes the design and implementation of the AuthLoop protocol.

5.1 Design Considerations

Before describing the full protocol, this section briefly discusses the design considerations that led to the AuthLoop authentication protocol. As previously mentioned, we are constrained in that there is no fully-fledged Public Key Infrastructure, meaning that Verifiers (i.e., end users) do not universally possess a strong credential. Moreover, because we are limited to transmission over the audio channel, the AuthLoop protocol must be highly bandwidth efficient.

The most natural choice for AuthLoop would be to reuse an authentication protocol such as Needham-Schroeder [50]. Reusing well-understood security protocols has great value. However, Needham-Schroeder is inappropriate because it assumes that both sides have public/private key pairs or can communicate with a third party for session key establishment. Goal G1 is therefore not practically achievable in real telephony systems if Needham-Schroeder is used. This protocol is also unsuitable as it does not establish session keys, meaning that achieving G2 would require frequent re-execution of the entire authentication protocol, which is likely to be highly inefficient.

TLS can achieve goals G1 and G2, and already does so for a wide range of traditional applications on the Web. Unfortunately, the handshaking and negotiation phases of TLS 1.2 require significant bandwidth. As we demonstrate in Section 3, unmodified use of this protocol can require an average of 97 seconds before authentication can be completed. However, because it can achieve goals G1 and G2, TLS 1.2 is useful as a template for our protocol, and we discuss what could be considered a highly-optimized version below. We note that while TLS 1.3 provides great promise for reducing handshaking costs, the current draft version requires more bandwidth than the AuthLoop protocol.

5.2 Protocol Definition

Figure 5 provides a formal definition for our authentication protocol. We describe this protocol below, and provide details about its implementation and parameterization (e.g., algorithm selection) in Section 5.4.

The AuthLoop protocol begins immediately after a call is terminated.[5] Either party, the Prover P (e.g., a call center) or the Verifier V (e.g., the end user) can initiate the call. V then transmits its identity (i.e., phone number) and a nonce N_V to P. Upon receiving this message, P transmits a nonce N_P, its certificate C_P, and signs the contents of the message to bind the nonce to its identity. Its identity, P, is transmitted via Caller ID and is also present in the certificate.

V then generates a pre-master secret S, and uses S to generate a session key k, which is the result of $HMAC(S, N_P, N_V)$. V then extracts P's public key from the certificate, encrypts S using that key and then computes $HMAC(k, \text{`}VRFY\text{'}, \#1, \#2)$, where 'VRFY' is a literal string, and #1 and #2 represent the contents of messages 1 and 2. $V$ then sends $S$ and the HMAC to $P$. $P$ decrypts the pre-master secret and uses it to similarly calculate $k$, after which is calculates $HMAC(k, \text{`}PROV\text{'}, \#1, \#2)$, which it then returns to V.

At this time, P has demonstrated knowledge of the private key associated with the public key included in its certificate, thereby authenticating the asserted identity. If the Prover does not provide the correct response, its claim of the Caller ID as its identity is rejected. Security goal G1 is therefore achieved. Moreover, P and V now share a session key k, which can be subsequently used to provide continued and efficient proofs (i.e., HMACs over incrementing nonces) that they remain on the call, thereby achieving Goal G2.

We note that the session key generation step between messages 2 and 3 can be extended to provide keys for protecting confidentiality and integrity (as is done in most TLS sessions). While these keys are not of value for voice communications (given the narrow bitrate of our channel), they can be used to protect client authentication credentials. We discuss this in greater detail in Section 7.

5.3 Formal Verification

We believe that our protocol is secure via inspection. However, to provide stronger guarantees, we use the Proverif v1.93 [22] automatic cryptographic protocol verifier to reason about the security of the AuthLoop handshake. Proverif requires that protocols be rewritten as Horn clauses and modeled in Pi Calculus, from which it can then reason about secrecy and authentication in the Dolev-Yao setting. AuthLoop was represented by a total of 60 lines of code, and Proverif verified the secrecy of the session key k. Further details about configuration will be available in our technical report.

Table 2: Authloop Message Sizes

Message Field	Size(Bits)
Verifier Hello	**144**
Nonce	96
Cert Ident Number	40
Protocol Command	8
Prover Hello	**1692**
Nonce	96
Certificate (optional)	1592
Protocol Command	8
Verifier Challenge	**1312**
Encrypted Premaster Secret	1224
HMAC	80
Protocol Command	8
Prover Response	**88**
HMAC	80
Protocol Command	8
Total With Certificate	**3236**
Total Without Certificate	**1648**

5.4 Implementation Parameters

Table 2 provides accounting of every bit used in the AuthLoop protocol for each message. Given the tight constraints on the channel, we use the following parameters and considerations to implement our protocol as efficiently as possible while still providing strong security guarantees.

We use elliptic curve cryptography for public key primitives. We used the Pyelliptic library for Python [9], which is a Python wrapper around OpenSSL. Keys were generated on curve `sect283r1`, and keys on this curve provide security equivalent to RSA 3456 [56]. For keyed hashes, we use SHA-256 as the underlying hash function for HMACs. To reduce transmission time, we compute the full 256-bit HMAC and truncate the result to 80 bits. Because the security factor of HMAC is dependent almost entirely on the length of the hash, this truncation maintains a security factor of 2^{-80} [21]. This security factor is a commonly accepted safe value [49] for the near future, and as our data transmission improves, the security factor can increase as well.

While similar to TLS 1.2, we have made a few important changes to reduce overhead. For instance, we do not perform cipher suite negotiation in every session and instead assume the default use of AES256_GCM and SHA256. Our link layer header contains a bit field indicating whether negotiation is necessary; however, it is our belief that starting with strong defaults and negotiating in the rare scenario where negotiation is necessary is critical to saving bandwidth for AuthLoop. Similarly, we are able to exclude additional optional information (e.g.,

[5] This is the telephony term for "delivered to its intended destination," and signifies the beginning of a call, not its end.

compression types supported) and the rigid TLS Record format to ensure that our overhead is minimized.

We also limit the contents of certificates. Our certificates consist of a protocol version, the prover's phone number, claimed identification (i.e., a name), validity period, unique certificate identification number, the certificate owner's ECC public key and a signature. Because certificate transmission comprises nearly half of the total transmission time, we implemented two variants of AuthLoop: the standard handshake and a version with a verifier-cached certificate. Certificate caching enables a significantly abbreviated handshake. For certificate caching, we include a 16-bit certificate identifier that the verifier sends to the prover to identify which certificate is cached. We discuss how we limit transmitted certificate chain size to a single certificate in Section 7.

Finally, we keep the most security-sensitive parameters as defined in the TLS specification, including recommended sizes for nonces (96 bits).

While our protocol implementation significantly reduces the overhead compared to TLS 1.2 for this application, there is still room for improvement. In particular, the encrypted pre-master secret requires 1224 bits for the 256-bit premaster secret. This expansion is due to the fact that while RSA has a simple primitive for direct encryption of a small value, with ECC one must use a hybrid encryption model called the Integrated Encryption Scheme (IEC), so a key must be shared separately from the encrypted data. Pyelliptic also includes a SHA-256 HMAC of the ECC keyshare and encrypted data to ensure integrity of the message (which is standard practice in IEC). Because the message already includes an HMAC, in future work we plan to save 256 bits (or 15% of the cached certificate handshake) by including the HMAC of the ECC share into the message HMAC.

6 Evaluation

Previous sections established the need for a custom authentication protocol using a voice channel modem to provide end-to-end authentication for telephone calls. In this section, we describe and evaluate our prototype implementation. In particular, we characterize the error performance of the modem across several audio codecs, compute the resulting actual throughput after layer 2 effects are taken into account, and finally measure the end to end timing of complete handshakes.

6.1 Prototype Implementation

Our prototype implementation consists of software implementing the protocol, link layer, and modem running on commodity PCs. While we envision that AuthLoop will eventually be a stand-alone embedded device or implemented in telephone hardware/software, a PC served as an ideal prototyping platform to evaluate the system.

Table 3: Bit Error Rates

Codec	Average Bit Error	Std. Dev
G.711	0.0%	0.0%
AMR-NB	0.3%	0.2%
Speex	0.5%	5%

We implemented the AuthLoop protocol in Python using the Pyelliptic library for cryptography. We also implemented the link layer in Python. Our modem was written in Matlab, and that code is responsible for modulating data, demodulating data, and sending and receiving samples over the voice channel. We used the Python Engine for Matlab to integrate our modem with Python. Our choice of Matlab facilitated rapid prototyping and development of the modem, but the Matlab runtime placed a considerable load on the PCs running the prototype. Accordingly, computation results, while already acceptable, should improve for embedded implementations.

We evaluate the modem and handshake using software audio channels configured to use one of three audio codecs: G.711 (μ-law), Adaptive MultiRate Narrow Band (AMR-NB), and Speex. These particular codecs were among the most common codecs used for landline audio compression, cellular audio, and VoIP audio, respectively. We use the `sox`[10] implementations of G.711 and AMR-NB and the `ffmpeg`[8] implementation of Speex. We use software audio channels to provide a common baseline of comparison, as no VoIP client or cellular device supports all of these codecs.

As link layer performance depends only on the bit error characteristics of the modem, we evaluate the link layer using a software loopback with tunable loss characteristics instead of a voice channel. This allowed us to fully and reproducibly test and evaluate the link layer.

6.2 Modem Evaluation

The most important characteristic of the modem is its resistance to bit errors. To measure bit error, we transmit 100 frames of 2000 random bits[6] each and measure the bit error after reception.

Table 3 shows the average and standard deviation of the bit error for various codecs. The modem saw no bit errors on the G.711 channel; this is reflective of the fact that G.711 is high-quality channel with very minimal processing and compression. AMR-NB and Speex

[6] 2000 bits was chosen as the first "round" number larger than the largest message in the AuthLoop handshake.

Table 4: Link Layer Transmission of 2000 bits

Bit Error Rate	Transmission Time	Goodput
0.1%	4.086 s (0.004)	490 bps
1%	6.130 s (0.009)	326 bps
2%	11.652 s (0.007)	172 bps

both saw minimal bit error as well, though Speex had a much higher variance in errors. Speex had such a high variance because one frame was truncated, resulting in a higher average error despite the fact the other 99 frames were received *with no error*.

6.3 Link Layer Evaluation

The most important characteristic of the link layer is its ability to optimize goodput – the actual amount of application data transmitted per unit time (removing overhead from consideration).

Table 4 shows as a function of bit error the transmission time and the goodput of the protocol compared to the theoretical optimal transmission time and goodput. The optimal numbers are computed from the optimal bit time (at 500 bits per second) plus 40ms of header and footer. The experimental numbers are the average of transmission of 50 messages with 2000 bits each. The table shows that in spite of high bit error rates (up to 2%) the link layer is able to complete message transmission. Of course, the effect of bit errors on goodput is substantial at larger rates. Fortunately, low bit error rates (e.g. 0.1%) result in a minor penalty to goodput – only 5bps lower than the optimal rate. Higher rates have a more severe impact, resulting in 65.8% and 34.7% of optimal goodput for 1% and 2% loss. Given our observations of bit error rates at less than 0.5% for all codecs, these results demonstrate that our Link Layer retransmission parameters are set with an acceptable range.

6.4 Handshake Evaluation

To evaluate the complete handshake, we measure the complete time from handshake start to handshake completion from the verifier's perspective. We evaluate both variants of the handshake: with and without the prover sending a certificate. Handshakes requiring a certificate exchange will take much longer than handshakes without a certificate. This is a natural consequence of simply sending more data.

Table 5 shows the total handshake times for calls over each of the three codecs. These results are over 10 calls each. Note that these times are corrected to remove the effects of instrumentation delays and artificial delays caused by IPC among the different components of our prototype that would be removed or consolidated in deployment.

From the verifier perspective, we find that cached-certificate exchanges are quite fast – averaging 4.844 seconds across all codecs. When certificates are not cached, our overall average time is 8.977 seconds. Differences in times taken for certificate exchanges for different codecs are caused by the relative underlying bit error rate of each codec. G.711 and Speex have much lower error rates than AMR-NB, and this results in a lower overall handshake time. In fact, because those codecs saw no errors during the tests, their execution times were virtually identical.

Most of the time spent in the handshake is spent in transmitting messages over the voice channel. In fact, transmission time accounts for *99%* of our handshake time. Computation and miscellaneous overhead average to less than 50 milliseconds for all messages. This indicates that AuthLoop is computationally minimal and can be implemented on a variety of platforms.

7 Discussion

This section provides a discussion of client authentication, public key infrastructure, and deployment considerations for AuthLoop.

7.1 Client Credentials

Up until this point, we have focused our discussion around strong authentication of one party in the phone call (i.e., the Prover). However, clients already engage in a weaker "application-layer" authentication when talking to many call centers. For instance, when calling a financial institution or ISP, users enter their account number and additional values including PINs and social security numbers. Without one final step, our threat model would allow for an adversary to successfully steal such credentials as follows: An adversary would launch a 3-Way call to both the victim client and the targeted institution. After passively observing the successful handshake, the adversary could capture the client's credentials (i.e., DTMF tone inputs) and hang up both ends of the call. The adversary could then call the targeted institution back spoofing the victim's Caller ID and present the correct credentials.

One of the advantages of TLS is that it allows for the generation of multiple session keys, for use not only in continued authentication, but also in the protection of data confidentiality and integrity. AuthLoop is no different. While the data throughput enabled by our modem is low, it is sufficiently large enough to carry encrypted copies of client credentials. Accordingly, an adversary attempting to execute the above attack would be unable to do so successfully because this sensitive information

Table 5: Handshake completion times

Codec	Cached Certificate	Certificate Exchanged
G.711	4.463 s (0.000)	8.279 s (0.000)
AMR-NB	5.608 s (0.776)	10.374 s (0.569)
Speex	4.427 s (0.000)	8.279 s (0.000)
Average	4.844 s	8.977 s

could easily be passed through AuthLoop (and therefore useless in a second session). Moreover, because users are already accustomed to entering such information when interacting with these entities, the user experience could continue without any observable difference.

7.2 Telephony PKI

One of the most significant problems facing SSL/TLS is its trust model. X.509 certificates are issued by a vast number of Certificate Authorities (CAs), whose root certificates can be used to verify the authenticity of a presented certificate. Unfortunately, the unregulated nature of who can issue certificates to whom (i.e., what authority does X have to verify and bind names to entity Y?) and even who can act as a CA have been known since the inception of the current Public Key Infrastructure [37]. This weakness has lead to a wide range of attacks, and enabled both the mistaken identity of domain owners and confusion as to which root-signed certificate can be trusted. Traditional certificates present another challenge in this environment - the existence of long verification chains in the presence of the bitrate limited audio channel means that the blind adoption of the Internet's traditional PKI model will simply fail if applied to telephony systems. As we demonstrated in our experiment in Table 1, transmitting the entirety of long certificate chains would simply be detrimental to the performance of AuthLoop.

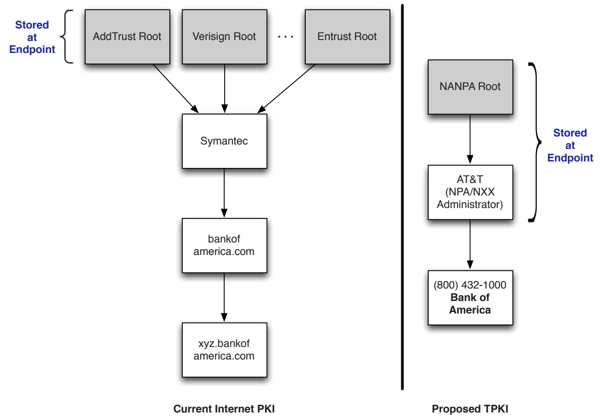


Figure 6: The Telephony Public Key Infrastructure (TPKI). Unlike in Internet model, the TPKI has a single root (NANPA) which is responsible for all block allocation, and a limited second level of CLECs who administer specific numbers. Accordingly, only the certificate for the number claimed in the current call needs to be sent during the handshake.

The structure of telephony networks leads to a natural, single rooted PKI system. Competitive Local Exchange Carriers (CLECs) are assigned blocks of phone numbers by the North American Numbering Plan Association (NANPA), and ownership of these blocks is easily confirmed through publicly posted resources such as NPA/NXX databases in North America. A similar observation was recently made in the secure Internet routing community, and resulted in the proposal of the Resource Public Key Infrastructure (RPKI) [45]. The advantage to this approach is that because all allocation of phone numbers is conducted under the ultimate authority of NANPA, all valid signatures on phone numbers must ultimately be rooted in a NANPA certificate. This Telephony Public Key Infrastructure (TPKI) reduces the length of certificate chains and allows us to easily store the root and all CLEC certificates in the US and associated territories (≈ 700 [46]) in just over 100 KiB of storage (1600 bits per certificate × 700). Alternatively, if certificates are only needed for toll free numbers, a single certificate for the company that administers all such numbers (i.e., Somos, Inc.) would be sufficient.

Figure 6 shows the advantages of our approach. Communicating with a specific server (xyz.bankofamerica.com) may require the transmission of three or more certificates before identity can be verified. Additionally, the existence of different roots adds confusion to the legitimacy of any claimed identity. Our proposed TPKI relies on a single NANPA root, and takes advantage of the relatively small total number of CLECs to require only single certificate for the calling number to be transmitted during the handshake. We leave further discussion of the details of the proposed TPKI (e.g., revocation, etc) to our future work.

7.3 Deployment Considerations

As our experiments demonstrate that AuthLoop is bandwidth and not processor bound, we believe that these techniques can be deployed successfully across a wide range of systems. For instance, AuthLoop can be embedded directly into new handset hardware. Moreover, it can be used immediately with legacy equipment through external adapters (e.g., Raspberry Pi). Alternatively, AuthLoop could be loaded onto mobile devices through a software update to the dialer, enabling large numbers of devices to immediately benefit.

Full deployments have the opportunity to make audio signaling of AuthLoop almost invisible to the user. If AuthLoop is in-line with the call audio, the system can remove AuthLoop transmissions from the audio sent to the user. In other words, users will never hear the AuthLoop handshakes or keep-alive messages. While our current strategy is to minimize the volume of the signaling so as to not interrupt a conversation (as has been done in other signaling research [62]), we believe that the in-line approach will ultimately provide the greatest stability and least intrusive user experience.

Lastly, we note that because AuthLoop is targeted across all telephony platforms, a range of security indicators will be necessary for successfully communicating authenticated identity to the user. However, given the limitations of space and the breadth of devices and their interfaces, we leave this significant exploration to our future work.

8 Conclusions

Phone systems serve as the trusted carriers of some of our most sensitive communications. In spite of this trust, authentication between two end points across this heterogeneous landscape was previously not possible. In this paper, we present AuthLoop to address this challenge. We began by designing a modem and supporting link layer protocol for the reliable delivery of data over a voice channel. With the limitations of this channel understood, we then presented a security model and protocol to provide explicit authentication of an assertion of Caller ID, and discussed ways in which client credentials could be subsequently protected. Finally, we demonstrated that AuthLoop reduced execution time by over an order of magnitude on average when compared to the direct application of TLS 1.2 to this problem. In so doing, we have demonstrated that end-to-end authentication is indeed possible across modern telephony networks.

Acknowledgment

The authors would like to thank our anonymous reviewers for their helpful comments and colleagues at the Florida Institute for Cybersecurity Research for their assistance in preparing this manuscript.

This work was supported in part by the US National Science Foundation under grant numbers CNS-1617474, CNS-1526718 and CNS-1464088. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] RedPhone. https://play.google.com/store/apps/details?id=org.thoughtcrime.redphone.

[2] Directory of Unknown Callers. http://www.800notes.com/, 2015.

[3] GSMK CryptoPhone. http://www.cryptophone.de/en/, 2015.

[4] Nomorobo. https://www.nomorobo.com/, 2015.

[5] PGPfone - Pretty Good Privacy Phone. http://www.pgpi.org/products/pgpfone/, 2015.

[6] Signal. https://itunes.apple.com/us/app/signal-private-messenger/id874139669?mt=8, 2015.

[7] Silent Circle. https://www.silentcircle.com/, 2015.

[8] ffmpeg. https://www.ffmpeg.org, 2016.

[9] Pyelliptic. https://pypi.python.org/pypi/pyelliptic, 2016.

[10] sox. http://sox.sourceforge.net/Main/HomePage, 2016.

[11] 3rd Generation Partnership Project. A Guide to 3rd Generation Security. Technical Report 33.900 version 1.2.0, 2000.

[12] 3rd Generation Partnership Project. 3G Security Principles and Objectives (3GPP TS 33.120). 2001.

[13] 3rd Generation Partnership Project. 3GPP TS 23.228 IP Multimedia Subsystem (IMS). (Release 11), 2012.

[14] D. Akhawe, B. Amann, M. Vallentin, and R. Sommer. Here's my cert, so trust me, maybe? Understanding TLS errors on the web. In *Proceedings of the 22nd International Conference on World Wide Web (WWW)*, pages 59–70, 2013.

[15] D. Akhawe and A. P. Felt. Alice in Warningland: A large-scale field study of browser security warning effectiveness. In *Proceedings of the USENIX Security Symposium*, 2013.

[16] F. Alegre, G. Soldi, and N. Evans. Evasion and obfuscation in automatic speaker verification. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 749–753, 2014.

[17] F. Alegre and R. Vipperla. On the vulnerability of automatic speaker recognition to spoofing attacks with artificial signals. In *Proceedings of the 20th European Signal Processing Conference (EUSIPCO)*, pages 36–40, 2012.

[18] V. Balasubramaniyan, A. Poonawalla, M. Ahamad, M. Hunter, and P. Traynor. PinDr0p: Using Single-Ended Audio Features to Determine Call Provenance. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2010.

[19] E. Barkan, E. Biham, and N. Keller. Instant ciphertext-only cryptanalysis of GSM encrypted communication. *Journal of Cryptology*, 21(3):392–429, 2008.

[20] A. Bates, J. Pletcher, T. Nichols, B. Hollembaek, and K. R. Butler. Forced perspectives: Evaluating an SSL trust enhancement at scale. In *Proceedings of the 2014 Internet Measurement Conference (IMC)*, pages 503–510. ACM, 2014.

[21] M. Bellare. New Proofs for NMAC and HMAC Security without Collision-Resistance. Advances in Cryptology - CRYPTO '06, 2006.

[22] B. Blanchet. ProVerif: Cryptographic protocol verifier in the formal model. `http://www.proverif.ens.fr/`, 2016.

[23] H. K. Bokharaei, A. Sahraei, Y. Ganjali, R. Keralapura, and A. Nucci. You can SPIT, but you can't hide: Spammer identification in telephony networks. In *Proceedings of the IEEE INFOCOM*, pages 41–45, 2011.

[24] R. Bresciani. The ZRTP protocol analysis on the Diffie-Hellman mode. *Foundations and Methods Research Group*, 2009.

[25] R. Bresciani, S. Superiore, S. Anna, and I. Pisa. The ZRTP protocol security considerations. Technical Report LSV-07-20, 2007.

[26] Y. J. Choi and S. J. Kim. An improvement on privacy and authentication in GSM. In *Proceedings of Workshop on Information Security Applications (WISA)*, 2004.

[27] J. Clark and P. C. Van Oorschot. SoK: SSL and HTTPS: Revisiting past challenges and evaluating certificate trust model enhancements. In *Proceedings of the IEEE Symposium on Security and Privacy (S&P)*, pages 511–525, 2013.

[28] Communications Fraud Control Association (CFCA). 2013 Global Fraud Loss Survey. `http://www.cvidya.com/media/62059/global-fraud_loss_survey2013.pdf`, 2013.

[29] I. Dacosta, M. Ahamad, and P. Traynor. Trust No One Else: Detecting MITM Attacks Against SSL/TLS Without Third-Parties. In *Proceedings of the European Symposium on Research in Computer Security (ESORICS)*, 2012.

[30] I. Dacosta, V. Balasubramaniyan, M. Ahamad, and P. Traynor. Improving Authentication Performance of Distributed SIP Proxies. *IEEE Transactions on Parallel and Distributed Systems (TPDS)*, 22(11):1804–1812, 2011.

[31] I. Dacosta and P. Traynor. Proxychain: Developing a Robust and Efficient Authentication Infrastructure for Carrier-Scale VoIP Networks. In *Proceedings of the USENIX Annual Technical Conference (ATC)*, 2010.

[32] R. Dhamija, J. D. Tygar, and M. Hearst. Why phishing works. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems (CHI)*, CHI '06, New York, NY, USA, 2006. ACM.

[33] A. Dhananjay, A. Sharma, M. Paik, J. Chen, T. K. Kuppusamy, J. Li, and L. Subramanian. Hermes: Data transmission over unknown voice channels. In *Proceedings of the Sixteenth Annual International Conference on Mobile Computing and Networking*, MobiCom, New York, NY, USA, 2010. ACM.

[34] Z. Durumeric, J. Kasten, D. Adrian, J. A. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann, J. Beekman, M. Payer, and V. Paxson. The matter of heartbleed. In *Proceedings of the 2014 Conference on Internet Measurement Conference (IMC)*, pages 475–488, New York, NY, USA, 2014. ACM.

[35] S. Egelman, L. F. Cranor, and J. Hong. You've been warned: An empirical study of the effectiveness of web browser phishing warnings. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI)*, 2008.

[36] C. Ellison, B. Frantz, B. Lampson, R. L. Rivest, B. Thomas, and T. Ylonen. SPKI Certificate Theory. IETF, RFC 2693, 1999.

[37] C. Ellison and B. Schneier. Ten risks of PKI: What you're not being told about public key infrastructure. *Computer Security Journal*, 16(1):1–7, 2000.

[38] R. Holz, L. Braun, N. Kammenhuber, and G. Carle. The SSL landscape: a thorough analysis of the x.509 PKI using active and passive measurements. In *Proceedings of the 2011 ACM SIGCOMM conference on Internet Measurement Conference (IMC)*, pages 427–444, 2011.

[39] L. S. Huang, A. Rice, E. Ellingsen, and C. Jackson. Analyzing forged SSL certificates in the wild. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, 2014.

[40] N. Jiang, Y. Jin, A. Skudlark, W.-L. Hsu, G. Jacobson, S. Prakasam, and Z.-L. Zhang. Isolating and analyzing fraud activities in a large cellular network via voice call graph analysis. In *Proceedings of the 10th international conference on Mobile systems, applications, and services (MobiSys)*, page 253, 2012.

[41] Q. Jin, A. R. Toth, A. W. Black, and T. Schultz. Is voice transformation a threat to speaker identification? In *Proceedings of the International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4845–4848. IEEE, 2008.

[42] N. N. Katugampala, K. T. Al-Naimi, S. Villette, and A. M. Kondoz. Real-time end-to-end secure voice communications over GSM voice channel. *2005 European Signal Processing Conference*, pages 1–4, 2005.

[43] P. Koopman and T. Chakravarty. Cyclic redundancy code (CRC) polynomial selection for embedded networks. In *2004 International Conference on Dependable Systems and Networks*, pages 145–154, June 2004.

[44] C. Lee, M. Hwang, and W. Yang. Enhanced privacy and authentication for the global system for mobile communications. *Wireless Networks*, 5(4):231–243, 1999.

[45] M. Lepinski, R. Barnes, and S. Kent. An Infrastructure to Support Secure Internet Routing. IETF, RFC 6480, 2012.

[46] Local Search Association. CLEC Information. `http://www.thelsa.org/main/clecinformation.aspx`, 2016.

[47] B. Mathieu, S. Niccolini, and D. Sisalem. SDRS: A Voice-over-IP spam detection and reaction system. *IEEE Security & Privacy Magazine*, 6(6):52–59, nov 2008.

[48] B. Moeller and A. Langley. TLS Fallback Signaling Cipher Suite Value (SCSV) for Preventing Protocol Downgrade Attacks. Internet-draft, Internet Engineering Task Force, 2014.

[49] National Institute of Standards and Technology. NIST Special Publication 800-107 Revision 1: Recommendation for Applications Using Approved Hash Algorithms. `http://csrc.nist.gov/publications/nistpubs/800-107-rev1/sp800-107-rev1.pdf`, 2008.

[50] R. Needham and M. Schroeder. Using encryption for authentication in large networks of computers. *Communications of the ACM*, 21(12):993–999, 1978.

[51] M. A. Ozkan, B. Ors, and G. Saldamli. Secure voice communication via GSM network. *2011 7th International Conference on Electrical and Electronics Engineering (ELECO)*, pages II–288–II–292, 2011.

[52] M. Petraschek, T. Hoeher, O. Jung, H. Hlavacs, and W. Gansterer. Security and usability aspects of Man-in-the-Middle attacks on ZRTP. *Journal of Universal Computer Science*, 14(5):673–692, 2008.

[53] A. Ramirez. Theft through cellular 'clone' calls. `http://www.nytimes.com/1992/04/07/business/theft-through-cellular-clone-calls.html`, April 7, 1992.

[54] B. Reaves, E. Shernan, A. Bates, H. Carter, and P. Traynor. Boxed Out: Blocking Cellular Interconnect Bypass Fraud at the Network Edge. In *Proceedings of the USENIX Security Symposium (SECURITY)*, 2015.

[55] E. Rescorla. *SSL and TLS: Designing and Building Secure Systems*. Addison-Wesley, 2001.

[56] C. Research. SEC 2: Recommended Elliptic Curve Domain Parameters, January 2010.

[57] R. Rivest and B. Lampson. SDSI: A Simple Distributed Security Infrastructure. `http://research.microsoft.com/en-us/um/people/blampson/59-sdsi/webpage.html`, 1996.

[58] S. Rosset, U. Murad, E. Neumann, Y. Idan, and G. Pinkas. Discovery of fraud rules for telecommunications-challenges and solutions. In *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 409–413, New York, NY, USA, 1999.

[59] D. Samfat, R. Molva, and N. Asokan. Untraceability in mobile networks. In *Proceedings of the First Annual International Conference on Mobile Computing and Networking (MobiCom)*, pages 26–36, 1995.

[60] S. E. Schechter, R. Dhamija, A. Ozment, and I. Fischer. The emperor's new security indicators. In *Proceedings of the IEEE Symposium on Security and Privacy (SP)*, 2007.

[61] H. Sengar. VoIP Fraud : Identifying a wolf in sheep's clothing. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 334–345, 2014.

[62] M. Sherr, E. Cronin, S. Clark, and M. Blaze. Signaling vulnerabilities in wiretapping systems. *IEEE Security & Privacy Magazine*, 3(6):13–25, November 2005.

[63] M. Shirvanian and N. Saxena. Wiretapping via mimicry: Short voice imitation man-in-the-middle attacks on crypto phones. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (CCS)*, pages 868 – 879.

[64] Sklar, Bernard. *Digital Communications: Fundamentals and Applications*. Prentice Hall, Upper Saddle River, N.J, second edition, Jan. 2001.

[65] J. Sobey, R. Biddle, P. van Oorschot, and A. S. Patrick. Exploring user reactions to new browser cues for extended validation certificates. In *Proceedings of the European Symposium on Research in Computer Security (ESORICS)*, 2008.

[66] Y. Stylianou. Voice transformation: A survey. In *Proceedings of the IEEE Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2009.

[67] TelTech. SpoofCard. `http://www.spoofcard.com/`, 2015.

[68] M. Toorani and A. Beheshti. Solutions to the GSM security weaknesses. In *Proceedings of the Second International Conference on Next Generation Mobile Applications, Services, and Technologies (NGMAST)*, pages 576–581, 2008.

[69] P. Traynor, P. McDaniel, and T. La Porta. *Security for Telecommunications Networks*. Number 978-0-387-72441-6 in Advances in Information Security Series. Springer, August 2008.

[70] A. Tyrberg. *Data Transmission over Speech Coded Voice Channels*. Master's Thesis, Linkoping University, 2006.

[71] Z. Wu, A. Khodabakhsh, C. Demiroglu, J. Yamagishi, D. Saito, T. Toda, and S. King. SAS: A speaker verification spoofing database containing diverse attacks. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4440–4444, 2015.

[72] Z. Wu and H. Li. Voice conversion and spoofing attack on speaker verification systems. In *Proceedings of the Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*. IEEE, 2013.

[73] P. Zimmermann. Zfone. `http://zfoneproject.com/`, 2015.

[74] P. Zimmermann and A. Johnston. ZRTP: Media Path Key Agreement for Unicast Secure RTP. IETF, RFC 6189, 2011.

[75] T. Zoller. TLS & SSLv3 Renegotiation Vulnerability. `http://www.g-sec.lu/practicaltls.pdf`, 2009.