Accountable Wiretapping

-or-

I know they can hear you now

Adam Bates
University of Oregon

Kevin Butler
University of Oregon

Micah Sherr
Georgetown University

Clay Shields
Georgetown University

Patrick Traynor
Georgia Institute of Technology

Dan Wallach
Rice University

Abstract

In many democratic countries, CALEA[1] *wiretaps are used by law enforcement agencies to perform investigations and gather evidence for legal procedures. However, existing CALEA wiretap implementations are often engineered with the assumption that wiretap operators are trustworthy and wiretap targets do not attempt to evade the wiretap. Although it may be possible to construct more robust wiretap architectures by reengineering significant portions of the telecommunications infrastructure, such efforts are prohibitively costly. This paper instead proposes a lightweight* accountable wiretapping *system for enabling secure audits of existing CALEA wiretapping systems. Our proposed system maintains a tamper-evident encrypted log over wiretap events, enforces access controls over wiretap records, and enables privacy-preserving aggregate queries and compliance checks. We demonstrate using campus-wide telephone trace data from a large university that our approach provides efficient auditing functionalities while incurring only modest overhead. Based on publicly available wiretap reporting statistics, we conservatively estimate that our architecture can support tamper-evident logging for all of the United States' ongoing CALEA wiretaps using three commodity PCs.*

1. Introduction

Legally authorized wiretaps are conducted in every democratic country in the world. Generally approved by some external judicial process, sanctioned interception allows law enforcement to gather information about call activity related to *potentially* illicit activities. The information generated by this process is both extremely valuable and sensitive, making its protection of paramount importance.

Like any other process that creates or manages important data, the ability to audit wiretaps is critical. Verifying the correctness of such data not only gives the public better protection against abuse and greater confidence in the process, but also provides law enforcement agencies with stronger guarantees for their evidence. However, because the existence of a wiretap is itself a secret, providing verifiable evidence that legal interception was correctly conducted and logged is difficult.

In this paper, we propose an accountable wiretapping architecture that enhances wiretapping systems by adding tamper-evident records of wiretap events. In addition to the standard wiretap channel to law enforcement agencies (LEAs), we introduce ENCRYPTOR devices that interpose on the output of CALEA switches. ENCRYPTORs transmit encrypted wiretap records to an external wiretap *log storage provider*, referred to as the "LOG", that stores the encrypted wiretap data and performs on-demand audits over the encrypted records. The LOG allows auditors such as a supervising court, a justice department, or an NGO, to reconcile events captured by LEAs with those in the LOG. Our threat model considers three potential adversaries: (i) *the target of the wiretap* who employs known denial-of-service attacks to overwhelm the wiretap's resources [32], (ii) *a malicious employee* of the service provider who wishes to undetectably perform an unauthorized wiretap using the CALEA APIs, and (iii) *a dishonest* LOG that attempts to learn information about current and/or previous wiretaps. We demonstrate that under our reasonable assumptions, our auditing system detects the former two attacks and provides privacy mechanisms to limit the unauthorized exposure of wiretap information against the third.

This paper makes the following contributions:

- **First academic proposal for detecting attacks against the CALEA infrastructure:** A number of re-

[1] Somewhat confusingly, "CALEA" is often used to refer to both the U.S. law regarding wiretap requirements as well as the systems (utilized by several countries) that are used to conduct wiretapping.

cent papers have demonstrated potential vulnerabilities in the wiretapping infrastructure [31, 32]. However, research in this area is largely outside of the public's purview. Our work represents the first public effort to improve both confidence and accuracy of legal telephony interception.

- **Develops attacker model for accountable wiretapping:** We introduce a threat model for accountable wiretapping, and argue that existing wiretap collection and retention services do not adequately protect the integrity of wiretap records.
- **Introduces new protocols to enable trustworthy wiretap auditing:** We describe protocols for performing efficient auditing, reporting and compliance verification over encrypted wiretap records. In particular, we present protocols for protecting the confidentiality of wiretap records and providing tabulation and proofs of completeness to an auditor.
- **Develops minimal-impact retrofit for current interception systems:** We introduce a distributed architecture for auditable wiretapping that can be deployed with minimal effort and cost.
- **Evaluates implementation through extensive performance study:** We build a proof-of-concept implementation of our auditing infrastructure and conduct a range of performance benchmarks. We measure our system using anonymized call data from a major university. Based on the reported number of CALEA wiretaps [11], we estimate that our system could accommodate all U.S. wiretaps on 3 commodity machines.

2. Background

This paper makes the distinction between *lawfully authorized* surveillance and illegal eavesdropping. The latter is a much more widely understood threat: an eavesdropper surreptitiously observes some traffic and attempts to learn the content of the communication and/or the identities of the communicants. The problem of illicit eavesdropping has been well studied, and eavesdropping tools as well as countermeasures are readily available. In contrast, lawfully authorized wiretaps are subject to legal constraints. Courts, or in some instances a law enforcement agency, must approve of the wiretap.

Wiretaps can be divided into two classes: *Pen register/trap and trace* ("pen/trap") devices record call information such as call establishment, call disconnection, call waiting, call redirection, and dialed digits. Pen register devices record the metadata related to outgoing calls; trap and trace devices do the same for incoming calls. Neither system records call content. In the United States, the requirements for obtaining pen/trap wiretaps are much lower than that for content wiretaps: law enforcement agencies (LEAs) need only assert that metadata is pertinent to an ongoing investigation [41].

The second class, *Content wiretaps*, capture both call metadata and call content. In many jurisdictions, LEAs must convince a court that they have a reasonable belief that a target has committed a crime (i.e., "probable cause"). In the U.S., pen/trap orders are more common than full content wiretaps. For example, 20,899 pen/trap orders [42] and 386 content wiretap orders [11] were authorized in 2008. For both pen/trap and content wiretaps, the wiretapping is carried out by technicians employed by the service provider. Wiretap data are then transferred in real-time to a *listening post* maintained by a law enforcement agency.

Despite the importance of wiretap evidence to investigators and the courts, there are only a few publicly available and impartial studies of wiretap systems and architectures [32, 31, 18]. In part, this is because conducting an academic study of wiretap systems is complicated by legal constraints and business practices. Wiretap systems are often closed-source "black boxes" with little publicly available information, and manufacturers often provide technical specifications only to law enforcement organizations. In the U.S., the possession of wiretap equipment by non-law enforcement personnel is generally prohibited and punishable by up to five years in prison [41].

The inability to properly study deployed wiretap systems gives an advantage to those who wish to circumvent them; those who intend to illegally subvert a surveillance system are not usually constrained by the laws governing access to the wiretaps. Indeed, the limited amount of research that has looked at wiretap systems [31] and standards [32] has shown that existing wiretaps are vulnerable to unilateral countermeasures by the target of the wiretap, resulting in incorrect call records and/or omissions in audio recordings.

It has recently been argued [28] that wiretapping architectures can be made more robust by, for example, mandating that traffic flow through centralized "interception points" and imposing key escrow for all encrypted communication. There are significant technological and economic costs and risks [5, 18] associated with such a massive re-architecting of our communications networks. Modification would likely require billions of dollars. As legal wiretaps make up a small faction of all communications, re-engineering the system to improve wiretap capabilities is impractical and costly.

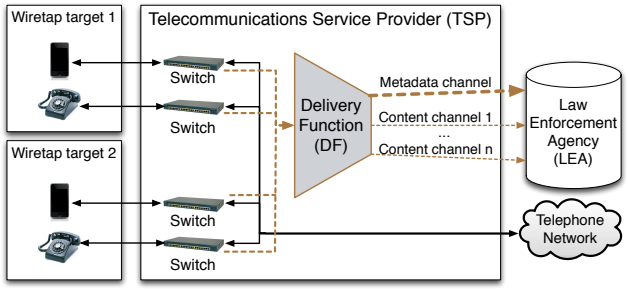

Figure 1: CALEA wiretap architecture. Solid arrows indicate standard telephony communication. Dashed arrows denote CALEA wiretap traffic flows.

2.1. CALEA Wiretap Architecture

The 1994 U.S. Communications Assistance for Law Enforcement Act (CALEA) [39] requires telecommunication service providers to incorporate wiretapping functionality into their switches. (Previous wiretap architectures relied on the physical cloning of wireline connections. The growth of cellular communication led to the promotion and eventual adoption of the newer and more flexible CALEA wiretapping architecture.) The CALEA law seeks to standardize methods and protocols for conveying wiretap information from providers' switches to law enforcement agencies. A major impetus of the Act was to standardize wiretap processing for new service offerings – in particular, cellular voice and data services. In 2003, telecommunication industry associations with input from law enforcement officials published ANSI Standard J-STD-025 (commonly referred to as the "J-Standard") [34], a collection of deployment guidelines and protocol specifications for communicating wiretap data to LEAs.

An overview of the CALEA wiretap architecture is presented in Figure 1.[2] Each subscribed service (e.g., landline phone, cellular, etc.) connects to the service provider through a switch located at the subscriber's *central office*.[3] The switch relays the user's communication to and from the rest of the telephone network (solid lines). If the subscriber is also a target of a wiretap, then the CALEA-compliant switch also sends a copy of the traffic to a *Delivery Function* (DF). Located at the service provider, the DF collects wiretap information from the switches and transfers it via J-Standard defined protocols to the LEA (dashed lines).

Call metadata (e.g., the identities of the communicants) are relayed via separate channels than call content to the LEA. The DF combines all sources of call metadata that are associated with a given wiretap, and encodes the information using the *Lawfully Authorized Electronic Surveillance Protocol* (LAESP) that is defined by the J-Standard. Metadata from different services (landline telephone, cellular, etc.) are multiplexed over the same *call metadata channel* between the DF and the LEA. LAESP protocol headers identify the pertinent wiretap order as well as the switch that captured the call event. The J-Standard also permits call metadata events belonging to different wiretap orders to be multiplexed together over the same call metadata channel, so long as they are associated with the same LEA.

In the case of content wiretaps, the DF creates one or more *call content channels* between itself and the LEA. Unlike the call metadata, call content is not multiplexed — each intercepted service is allocated with its own dedicated call content channel. Intercepted content communication is typically copied verbatim and transferred over the call content channel.

2.2. Vulnerabilities in CALEA Wiretaps

Previous work has shown that CALEA wiretaps are vulnerable to target-initiated DoS attacks [32]. Although the J-Standard does not mandate a particular type of connection between the DF and the LEA, it recommends using a 64kbps ISDN B line for each call content and call metadata channel. The target of the wiretap may overwhelm the 64kbps connection by purposefully inducing a high rate of call events. For example, in the case of content wiretaps, six LAESP messages are sent via the call metadata channel whenever the subject attempts and aborts a phone call, consuming 1.3kB of bandwidth per attempt. Telephony services which enable the target to attempt more than 6.2 calls per second (as is the case for VoIP services) permit the target to overwhelm the 64kbps capacity of the call metadata channel. Additionally, the target may apply other signaling techniques (e.g., rapid ISDN feature key selection) to similarly exhaust the call metadata channel. (Recall that the DF multiplexes signaling information from multiple services over the same call metadata channel, improving the target's ability to conduct DoS attacks.) Since the call metadata channel carries control messages which indicate the start and end times of audio content [34], its exhaustion may also lead to gaps in content recordings [32].

3. Accountable Wiretapping

There is little publicly available information regarding the operation and features of existing CALEA wiretap *implementations*. Manufacturers often closely guard the operation, security features, and limitations of their wiretap

[2] The *last-mile* connection between a mobile phone and the switch is considerably more complex. Conceptually, voice and data packets are routed via a Mobile Switching Center to the subscriber's central office. For clarity, Figure 1 omits these extra hops, which are orthogonal to the operation of the wiretap.

[3] The term central office is somewhat of a misnomer; service providers maintain many such central offices, and a typical central office may serve a few thousand customers.

systems. However, the product whitepapers and manuals that are accessible [9, 8] indicate that wiretap systems perform little logging. In part, this is by design: the *existence* of a wiretap is considered sensitive information. Preventing a (potentially rogue) technician from enumerating all past and present wiretaps reduces the risk that wiretaps become exposed.

This paper argues that wiretap events can be logged in a privacy-preserving manner, enabling secure audits of CALEA wiretaps. There are unfortunately no previously defined or well-established security requirements for conducting *auditable wiretaps*: neither the J-Standard nor any product literature of which we are aware describe technologies for conducting audits. Similarly, U.S. and European law do not appear to consider an audit process.

We note that at least one company advertises collection and retention services that can be used in a wiretap audit [43]. However, their product receives cleartext wiretap information from service providers, and is therefore inherently trusted with such data. In contrast, our architecture assumes a potentially untrusted logging service, and ensures that the LOG (i) never obtains access to plaintext wiretap records, (ii) cannot determine the number or scope of wiretaps, and (iii) can provide proofs to the auditors that it has correctly recorded all (encrypted) data. Given the sensitivity of wiretap data and the ease at which trusted systems may become compromised, we argue that the storage service should be modeled as an untrusted participant.

3.1. Security Goals

Our accountable wiretapping system aims to achieve the following *audit* goals:

Integrity**:** A wiretap audit should identify modified or corrupted wiretap records. Formally, if $L = \{\tau_j, \ldots, \tau_k\}$ are the wiretap records belonging to wiretap W that are sent by the service provider between times $[T_s, T_e]$, and $L' = \{\tau'_l, \ldots, \tau'_m\}$ are the wiretap records received by the LEA that purportedly belong to W within times $[T_s, T_e]$, then the audit should identify the *true* wiretap records $L \cap L'$.

Completeness**:** A wiretap audit should identify gaps in the records collected by the LEA. Given the above definitions of L and L', the audit should determine whether $L' \supseteq L$.

Date Compliance**:** A wiretap-granting authority authorizes a wiretap for a date range. The audit process should determine whether events were captured outside of that range. That is, given a date range $[T_s, T_e]$ and a wiretap event τ, the audit should reveal whether the *interception* of τ occurred within $[T_s, T_e]$.

Importantly, the audit goals are sufficient to detect the DoS attacks described in Section 2.2: An auditor can detect LAESP messages that have been corrupted or lost due to target-initiated signaling attacks by verifying the wiretap records' Integrity and Completeness.

Additionally, our audit process should support the efficient collection of information for legally mandated wiretap reporting requirements. The Omnibus Crime Control and Safe Streets Act [41] requires the Administrative Office of the U.S. Courts to present an annual wiretap report to Congress. The report includes the number of new and expiring pen/trap and content wiretaps.[4,5] European nations have similar reporting requirements (e.g., England's Regulation of Investigatory Powers Act of 2000 [6]). Towards achieving compliance with these reporting requirements, our architecture supports the following *accounting* goal:

Reporting**:** The audit should accurately report the number of new pen/trap wiretaps, new content wiretaps, expiring pen/trap wiretaps, and expiring content wiretaps within a specified time interval $[T_s, T_e]$.

By reconciling with law enforcement and court records, the Reporting functionality can also be applied to *detect unauthorized wiretaps* (i.e., a content or pen/trap wiretap that has not been authorized by a court or law enforcement agency, respectively). In addition to detection, an audit process that supports Reporting allows the auditor to issue repeated requests to discover the time that the illegal wiretap was initiated.

Our proposed system is a first step in developing more accountable wiretap systems. As such, we do not prevent attacks similar to the "Athens Affair" [24] in which telephony switch software was directly modified by an insider. We believe that our model is still powerful, especially considering that most other security infrastructure also fails when an adversary gains direct access to a machine. Additionally, our current system only considers pen/trap orders and not content wiretaps. As Pen registers represented 98% of total U.S. wiretap orders in 2008 [11, 42], our system already achieves our security audit goals for the vast majority of wiretaps.

3.2. Architecture and Participants

Our proposed audit process is designed to augment existing CALEA deployments without requiring significant (and costly) modifications at the service provider. In particular,

[4] Although U.S. law requires the Attorney General to separately report the number of pen/trap and content wiretaps, the number of pen/trap wiretaps has not regularly been disclosed.

[5] U.S. Public Law 106-197 additionally requires the AO to report the number of times that law enforcement detected that the target's communications were encrypted [40]. Interestingly, encryption does not seem to be widely applied: the 2010 Wiretap Report discloses only six instances of encryption, and notes that this "...did not prevent officials from obtaining the plain text of the communications." [12]

our solution introduces a single component — the **Encryptor** — at the provider's central office.

Recall that the Delivery Function (DF) collects wiretap information from various switches located at the central office, and transmits call metadata and (optionally) call content pertaining to a wiretap to a LEA (see Figure 1). In our accountable wiretapping architecture, the DF is configured to also send a copy of wiretap events to the ENCRYPTOR. Located at the service provider, the ENCRYPTOR encrypts wiretap events and transfers them to an off-site storage facility called the LOG (see Figure 2, Left). The LOG is an untrusted repository of encrypted wiretap events whose purpose is to archive wiretap data and assist in the audit process. Unlike commercial wiretap data retention stores [43] which share a similar architecture to our design, the LOG does not have access to plaintext wiretap information and cannot enumerate the wiretaps.

The LOG may store data from multiple wiretaps and possibly from multiple telecommunications service providers. By outsourcing the burden of wiretap audits away from service providers, the LOG assumes the responsibilities of data maintenance and retention while providing a small barrier to participate: From the perspective of the DF, the ENCRYPTOR operates as a sink, collecting call metadata and content as would a downstream LEA. In turn, the LOG operates as a database of auditable wiretap records. Although the LOG is not trusted, we argue below that our architecture ensures that insertions, modifications, or omissions of wiretap events will be detected during an audit.

Finally, we assume the existence of a trusted **Wiretap Authority** (e.g., a court) that authorizes the service provider to conduct the wiretap.

Audit Types. Our architecture provides support for two types of audits. In a *court audit*, the auditor queries the LOG for all records pertaining to a particular wiretap (see Figure 2, Center). For simplicity, we refer to this authority as a **court**, although in principle a court audit could be conducted by a non-judiciary body (e.g., a Justice Department). A court may be authorized to audit one or more wiretaps (for example, the wiretaps it previously issued), but cannot learn any information about the wiretaps for which the Wiretap Authority has not granted it access. Court audits may be appropriate, for example, to ensure (via the Integrity and Completeness properties) that all wiretap events were correctly captured by a LEA. Such audits are also useful to detect target-initiated DoS attacks against CALEA (see Section 2.2).

In an *accounting audit*, an **accountant** gathers statistics from the LOG regarding the total number of pen/trap and content wiretaps (see Figure 2, Right). The accountants are restricted to coarse-grained data: they should not learn any information regarding individual wiretap events during an accounting audit. However, the accountants can compile statistics over all the wiretaps stored at the LOG.

Note that both the court auditor and the accountant are trusted. In the former case, the court auditor *is* the judicial authority; if malicious, such an authority does not need to circumvent the wiretap audit in order to corrupt the judicial process. Similarly, since accountants are already tasked with compiling and reporting wiretap figures, they can simply provide false results. That is, the justice system already implicitly trusts the court to behave honestly and wiretap tabulators to report accurate statistics; our architecture assigns similar trust. As we describe next, we must still be able to detect any malicious behavior by the untrusted LOG.

3.3. Threat Model

The overarching goals of our accountable wiretapping infrastructure are tamper-evidence, compliance with reporting requirements, and privacy: Modifications, insertions, or omissions of wiretap records by the LOG should be detectable, and no auditor should have access to either coarse- or fine-grained wiretap information to which the Wiretap Authority has not explicitly granted it access.

We model three adversaries:

- **Wiretap Target:** The target of the wiretap may attempt to cause denial-of-service against the wiretap by overwhelming the call metadata channel between the Delivery Function (DF) and the LEA. The target can generate a high frequency of wiretap events (potentially using different devices), causing the resultant LAESP messages to consume the channel's bandwidth [32]. A goal of our architecture is to maintain a complete (Completeness) and accurate (Integrity) record of wiretap events, enabling court auditors to discover the resulting gaps in the transcripts collected by LEAs.

- **Unauthorized Wiretappers:** We provide some protection against attackers who provision unauthorized CALEA wiretaps. Such attackers may include rogue employees of the service provider or overzealous law enforcement officials. As discussed above, we do not protect against all forms of data exfiltration; unencrypted data flows throughout the phone system, and e2e solutions that protect against interception along these points would require a costly reengineering of the existing communications infrastructure. Instead, we focus on attackers who use the DF's built-in functionality to conduct unauthorized wiretaps. We assume that all wiretap events are captured by the ENCRYPTOR.

  An accountable wiretap system that achieves the Reporting property ensures that such unauthorized wiretapping will be detected during accounting audits,

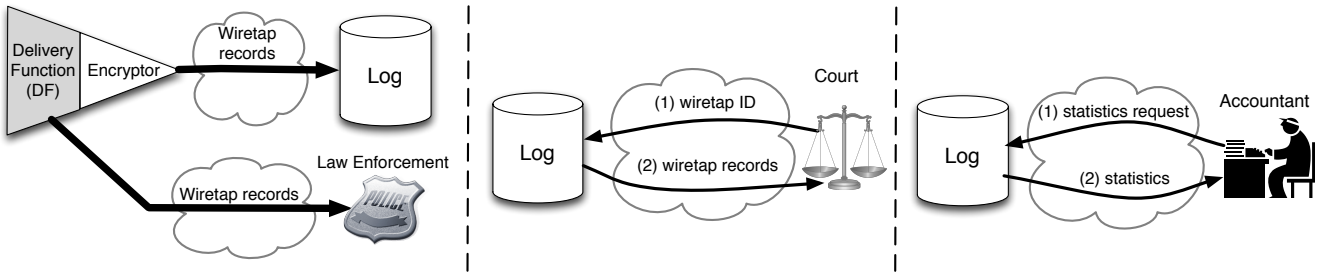

Figure 2: Wiretapping audit workflows. *Left:* The ENCRYPTOR transfers encrypted copies of wiretap records to the LOG. *Center:* A court auditor requests wiretap records from the Log. *Right:* An accountant gathers coarse-grained statistics about collected wiretaps.

since audit results can be reconciled with records maintained by the Wiretap Authority.

- **Dishonest Log:** As described above, the LOG receives encrypted wiretap records from the ENCRYPTOR. A dishonest LOG may attempt to corrupt court or accounting audits by modifying or deleting these records or by inserting false records. Additionally, the LOG may attempt to circumvent the privacy properties of our proposed architecture by either learning the plaintext of the encrypted wiretap records or by discovering timing information about wiretap events.

  Our architecture is designed to provide both *confidentiality* (the LOG cannot decipher encrypted records) and *unlinkability* (the LOG cannot determine that two encrypted records belong to the same wiretap). The former ensures the privacy of wiretap events, while the latter aggravates timing analysis.

Additional Network and Trust Assumptions. As discussed above, we assume that court and accountant auditors are correct and obey our protocols. (We do, however, prevent a court auditor from learning any information about wiretaps to which the Wiretap Authority has not granted it access.) We further assume that the service provider is honest. In particular, the DF generates actual wiretap events (either legally authorized or not) and the ENCRYPTOR behaves correctly and does not release key material. The ENCRYPTOR captures all wiretap events output by the DF. Ideally (but not necessarily), the connection between the ENCRYPTOR and the LOG should be adequately provisioned to prevent loss. As discussed in Section 4.3, our architecture detects such losses, but cannot distinguish between transmission loss and the LOG's purposeful omission of wiretap records. Finally, we assume that the ENCRYPTOR has an accurate clock.

4. Protocol

To protect the confidentiality of wiretap records, the ENCRYPTOR encrypts records with a symmetric key known only to itself and a *court auditor* (recall that the *accountant* is not authorized to access individual wiretap records). Additionally, to enable the court auditor to detect inserted or modified records, the ENCRYPTOR digitally signs each message before it is sent to the LOG. Handling omissions in a court audit is slightly less straightforward, and relies on the ENCRYPTOR's inclusion of encrypted sequence numbers as well as periodic heartbeat messages. A court auditor, who can decrypt the sequence numbers, can detect omissions between reported wiretap events by locating gaps in the sequence numbers. The use of regularly scheduled *heartbeat messages* bounds the LOG's ability to omit records at either end of the audit – that is, messages can be deleted, but the omission of the heartbeat indicates tampering, as do gaps that occur before or after the heartbeats.

To support accounting audits, the ENCRYPTOR encrypts counters along with each record using an additive homomorphic encryption scheme. Here, counters capture the number of new and expiring wiretaps, enabling the Reporting property. During an accounting audit, the LOG computes the sum over the counter ciphertexts and sends the result to the accountant. The accountant, who possesses the private key, can then decipher the result. The use of digital signatures and nonce summations (explained in Section 4.3) enables the auditor to verify the LOG's results.

4.1. Keying

The Wiretap Authority ("Authority") authorizes the service provider to conduct a wiretap, and is responsible for generating and disseminating cryptographic keys. To ensure authenticity, integrity, and non-repudiation of wiretap events, the Authority generates an *authenticity keypair* $\mathcal{A} = (A^+, A^-)$ and securely shares the signing key A^- with the ENCRYPTOR. For each wiretap, the Authority provides the ENCRYPTOR with a single symmetric key r (the

Key	Description	Assignment	Knowledge
r	*Record key*. Encrypts wiretap records (either call metadata or content).	Per wiretap	ENCRYPTOR, Court Auditor
A^+	*Public authenticity key*. Enables authenticity and integrity checking of wiretap records.	Per ENCRYPTOR	All parties / Public
A^-	*Private authenticity key*. Enables authenticity and integrity checking of wiretap records.	Per ENCRYPTOR	ENCRYPTOR
G^+	*Public aggregation key*. Encrypts wiretap statistics.	Per ENCRYPTOR	ENCRYPTOR
G^-	*Private aggregation key*. Decrypts encrypted wiretap statistics.	Per ENCRYPTOR	Accountant

Table 1: Summary of keys. The *Assignment* column indicates whether the key varies between wiretaps ("per wiretap") or is common to all wiretaps protected by the ENCRYPTOR ("per ENCRYPTOR"). The *Knowledge* column indicates the parties who possess the key. We denote symmetric and asymmetric keys respectively in lowercase and uppercase.

record key) used to encrypt wiretap events. The Authority also designates a *validity period* $T_{\text{start,stop}}$ over which the wiretap is authorized, and shares $T_{\text{start,stop}}$ with the ENCRYPTOR. Finally, the Authority creates an *aggregation keypair* $\mathcal{G} = (G^+, G^-)$ and gives G^+ to the ENCRYPTOR. The use of the aggregation keypair is explained in Section 4.2.

During the course of a court audit, the Wiretap Authority shares r with the court auditor. For accounting audits, the Authority shares G^- with the accountant. The public key A^+ used for verifying the ENCRYPTOR's signatures is public, and we assume all auditors have knowledge of this public key (e.g., through a certificate signed by the Authority). Note that the LOG does not know any of r, G^+, or G^-. A summary of the keys used by our protocol is presented in Table 1.

4.2. Event Logging

The ENCRYPTOR collects events from the Delivery Function (DF), encrypts them, and transmits the ciphertexts to the LOG. The ENCRYPTOR maintains minimal state, storing only the record key and the private authenticity and aggregation keys, as well as a per-wiretap event counter.

In this section, we describe our protocol for ensuring confidentiality and unlinkability (the LOG cannot discover the plaintext records, nor can it enumerate wiretaps or link records as belonging to the same wiretap). The correctness of audit results (i.e., Integrity, Completeness, Date Compliance, and Reporting) are argued in Section 5.

Let $\tau_1, \tau_2, \ldots$ be a continuous stream of wiretap events produced by the DF. Note that the stream may include events pertaining to multiple wiretaps. We let τ_i be the ith event of this sequence, and define t_i to be the time that the ENCRYPTOR received τ_i from the DF. Without loss of generality, we denote the wiretap that produced τ_i as w and the ENCRYPTOR's event counter for w to be I_w. The event counter I_w is incremented by one whenever the ENCRYPTOR transmits an event belonging to wiretap w.

The ENCRYPTOR prepares the message

$$M_i = \langle t_i \,||\, E_r(\tau_i \,||\, I_w \,||\, h(\tau_i||I_w)) \,||\, \mathcal{B}_i \rangle$$

where $||$ denotes concatenation, $E_k(Q)$ is the encryption of Q using symmetric key k, $h(\cdot)$ is a cryptographic checksum over its input, and $\mathcal{B}_i$ is an *aggregation block* which is described below. The ENCRYPTOR sends

$$\text{ENCRYPTOR} \rightarrow \text{Log} : \langle M_i \,||\, \sigma_{A^-}(M_i) \rangle$$

to the LOG, where $\sigma_{K^-}(Q)$ is a digital signature over message Q using key K^-. Upon receipt, the LOG stores the $\langle M_i, \sigma_{A^-}(M_i) \rangle$ message.

Aggregation Block. The aggregation block $\mathcal{B}$ is an encrypted set of counters that are used in accounting audits. Let $\mathcal{E}_{K^+}(Q)$ be an additive homomorphic encryption of message Q using public key K^+; that is, $\mathcal{E}_{K^-}(\mathcal{E}_{K^+}(Q_1) \oplus \mathcal{E}_{K^+}(Q_2)) = Q_1 \oplus Q_2$, where K^- is the private key and $\oplus$ is modular addition.

We perform multiple simultaneous homomorphic additions using a single ciphertext by partitioning the plaintext message Q into p elements $q_1, \ldots, q_p$; that is, $\texttt{sizeof}(Q) = \sum_{k=1}^{p} \texttt{sizeof}(q_i)$, where $\texttt{sizeof}(D)$ is the number of bits in D. Hence,

$$\mathcal{E}_{K^-}(\mathcal{E}_{K^+}(q_1 \,||\, \ldots \,||\, q_k) \oplus \mathcal{E}_{K^+}(q'_1 \,||\, \ldots \,||\, q'_k)) = (q_1 \oplus q'_1) \,||\, \ldots \,||\, (q_p \oplus q'_p) \quad (1)$$

where the values of $q_1 \oplus q'_1$ through $q_p \oplus q'_p$ are obtained by accessing the relevant "partition" of the result. Similarly, decryption of the above ciphertext gives the right-hand side of Equation 1, which can then be deconstructed into the fixed size partitions, revealing the sums $(q_1 \oplus q'_1), \ldots, (q_p \oplus q'_p)$.

Importantly, Equation 1 holds only if no summation $q_i \oplus q'_i$ overflows into the next higher ordered partition. In our implementation, we use the additive Paillier homomorphic encryption scheme [21] with a 1024-bit or 2048-bit modulus (i.e., 1024 and 2048-bit plaintexts). As described in Section 6.2, we partition this plaintext in a manner that allows us to store approximately 10 million entries before overflow can occur.

Given the above construction, we define the aggregation block $\mathcal{B}_i$ of wiretap event i ($i \geq 1$) as:

$$\mathcal{E}_{G^+}(\quad \texttt{1} \,||\, R_i \,||\, R_{i-1} \,||\, \text{newP}_i \,||\, \text{newC}_i \,||\, \text{expireP}_i \,||\, \text{expireC}_i \,)$$

where R_i is a large nonce and newP_i, newC_i, expireP_i, and expireC_i are `1` iff wiretap event i respectively corresponds to a new pen/trap wiretap, a new content wiretap, an expiring pen/trap wiretap, or an expiring content wiretap. Otherwise, the value of the counter is `0`. When $i = 1$, we define R_{i-1} to be `0`.

Conceptually, the newP_i, newC_i, expireP_i, and expireC_i counters allow an accountant to determine the number of new and expiring wiretap events within a specified date range. (Recall that the LOG performs the additions over the ciphertexts and returns the result to the accountant.) The inclusion of the nonces and the `1` constants enable the accountant to perform Completeness checks, and is explained in Section 5.

Special Message Types. The Wiretap Authority authorizes a wiretap for a given validity period $T_{\text{start,stop}}$. At the start of this period, the ENCRYPTOR generates a wiretap event where the content τ_i is the string "`start`". Similarly, at the end of the period, the ENCRYPTOR produces an event with the contents "`stop`". Both events are transmitted using the standard message format. (Note that in the former case, either newP_i or newC_i will be `1`, and in the latter case, either expireP_i or expireC_i will be `1`.)

Heartbeat Messages. The ENCRYPTOR periodically transmits a special *heartbeat* message for each wiretap. Injected according to a Poisson process, the heartbeats obscure the timing between consecutive events in a wiretap, and hence aggravate the LOG's ability to link (encrypted) events as belonging to the same wiretap. Additionally, the heartbeat message bounds the LOG's ability to omit entries for court and accounting audits. A heartbeat message H_w for a wiretap w is of the form $\langle t_i \mid\mid E_r(\texttt{heartbeat} \mid\mid I_w \mid\mid h(\texttt{heartbeat}||I_w)) \mid\mid \mathcal{B}_i\rangle$, where the newP_i, newC_i, expireP_i, and expireC_i counters in $\mathcal{B}_i$ are all `0`. Each heartbeat message uses a different wiretap's record key (r), selected in a round-robin fashion.

4.3. Audits

Our audit protocol serves three goals: to retrieve wiretap information from storage, to calculate aggregate statistics over the stored information, and to verify the authenticity, integrity, and accuracy of the results obtained from the LOG. Below, we describe the audit protocols for achieving the first two goals. We defer a security analysis of the audit protocols to Section 5.

Court Audits. A court audit retrieves all records for a specific wiretap from the LOG. To perform a court audit, the court auditor sends the LOG a request for all records between timestamps T_s and T_e:

$$\text{Court Auditor} \rightarrow \text{Log} : \texttt{CourtAudit}, T_s, T_e$$

A correct LOG then returns *all* records M_i where $T_s \leq t_i \leq T_e$ and t_i is the timestamp belonging to M_i:

$$\begin{aligned}\text{Log} \rightarrow \text{Court Auditor} : \langle & M_j, \sigma_{A^-}(M_j), \\ & M_{j+1}, \sigma_{A^-}(M_{j+1}) \ldots, M_k, \sigma_{A^-}(M_k)\rangle \end{aligned} \quad (2)$$

Given messages $M_j, M_{j+1}, \ldots, M_k$, the court auditor attempts to decrypt each M_i using the wiretap key r. Messages that cannot be decrypted (as indicated by a failure in matching the cryptographic checksum) are assumed to belong to a different wiretap (either as legitimate messages or inserted noise) and are discarded. Similarly, messages whose signature verification fails are ignored. The remaining records belong to the wiretap of interest, and gaps in sequence numbers (I_w) indicate missed messages.

Accounting Audits. An accounting audit should reveal the number of new and expiring wiretaps that occurred over a specified time period. To conduct an audit, the accountant transmits a request to the LOG:

$$\text{Accountant} \rightarrow \text{Log} : \texttt{AccountingAudit}, T_s, T_e$$

The LOG will then determine the wiretap event indices a and z ($a \leq z$) such the times corresponding to messages M_a and M_z (i.e., t_a and t_z) are the respective minimum and maximum times bound by $[T_s, T_e]$.

To calculate the statistics of interest, the LOG computes $\overline{\mathcal{B}}_{az} = \sum_{k=a}^{z} \mathcal{B}_k$. From the definition of $\mathcal{B}$ and Equation 1, we have

$$\begin{aligned}\overline{\mathcal{B}}_{az} = & \mathcal{E}_{G^+}\left(z - a + 1 \mid\mid \sum_{k=a}^{z} R_k \mid\mid \sum_{k=a}^{z} R_{k-1} \mid\mid \sum_{k=a}^{z} \text{newP}_k \mid\mid \right. \\ & \left. \sum_{k=a}^{z} \text{newC}_k \mid\mid \sum_{k=a}^{z} \text{expireP}_k \mid\mid \sum_{k=a}^{z} \text{expireC}_k\right)\end{aligned} \quad (3)$$

The LOG then returns the response:

$$\text{Log} \rightarrow \text{Accountant} : M_a, \sigma_{A^-}(M_a), M_z, \sigma_{A^-}(M_z), \overline{\mathcal{B}}_{az}$$

The accountant can then decrypt $\overline{\mathcal{B}}_{az}$ to obtain the total number of new pen/trap and content wiretaps as well as the number of expiring pen/trap and content wiretaps in $[T_s, T_e]$. Although the size of $\overline{\mathcal{B}}_{az}$ may be large due to the homomorphic additions, we note this cost is ephemeral — the LOG does not store $\overline{\mathcal{B}}_{az}$ and communication costs only occur during the (infrequent) audits.

5. Security Analysis

Given the importance of wiretap information to the evidence and investigative intelligence gathering processes, we

argue that there is a significant need to accurately *assess* the integrity and completeness of the collected information. In this section, we describe the trustworthiness of our accountable wiretapping infrastructure in the presence of adversaries who wish to either thwart the reliable collection of wiretap data, or conduct unauthorized interception.

5.1. Detecting Target-Initiated DoS Attacks

In the standard CALEA wiretap architecture, the target of the wiretap may cause LAESP messages to be lost in transit to the LEA by generating a flood of signaling information. Because each signal must be translated into a LAESP message, the under-provisioned connection between the DF and the LEA drops packets [32]. Since LAESP messages do not include sequence numbers [34], their loss may be undetected.

Our architecture supports the detection of lost LAESP messages through redundant storage and sequence numbering. During a court audit, wiretap records stored in the LOG may be reconciled with transcripts maintained by the LEAs to detect missed LAESP messages. LEAs can fill-in the missing pieces of their wiretap transcripts using data stored at the LOG. (Recall that since LOG messages are signed by the trusted ENCRYPTOR, such records are guaranteed to be authentic.)

Although the connection between the ENCRYPTOR and the LOG should be adequately provisioned, message loss may of course still occur. However, the use of sequence numbers enables the trivial detection and count of lost messages, achieving our Completeness goal.

5.2. Detecting Unauthorized Wiretaps

Our architecture detects the presence of certain unauthorized wiretaps. Notably, our solution does not protect against adversaries who can bypass the ENCRYPTOR. For example, a rogue employee of the service provider who can wiretap at various points throughout the telephone network can likely avoid detection. To mitigate such attacks, a more general data leakage prevention solution is required. In this paper, however, our focus is on reliable wiretap *audits*, and we briefly note that we can detect unauthorized wiretaps in the cases in which the intercepted data flows through an ENCRYPTOR. Such cases may occur, for instance, if an attacker is able to compromise a DF, but physically secured links require all outgoing communication from the DF to flow through the ENCRYPTOR.

Assuming that the unauthorized wiretap's data are relayed through the ENCRYPTOR, then the wiretap's presence will be revealed during accounting audits. That is, the accounting records will not reconcile with those of the Wiretap Authority, indicating the presence of the illegal wiretap. The Reporting property of our wiretap architecture allows an accountant to "hone in" on the time at which the illegal wiretap was provisioned; i.e., by conducting a binary search using multiple queries.

5.3. Protecting against a Malicious LOG

A LOG is an outsourced storage provider that receives wiretap information from potentially many DFs. Given the sensitivity of wiretap data, the LOG should not be trusted to have access to either individual wiretap records, nor should it be able to reliably discern coarse-grain data about past or ongoing wiretaps.

Confidentiality and Privacy. The confidentiality of wiretap records is protected through the straightforward use of encryption: The LOG does not have access to any private keys, and cannot decrypt wiretap events or aggregation blocks.

The LOG knows the time that each incoming message arrives, and can use this information to reason about the number of wiretaps and the potential association between any two encrypted wiretap events (for instance, that they belong to the same wiretap). However, heartbeat messages, which are inserted according to a random Poisson process, significantly hinder the LOG's ability to perform timing analysis. As we show in Section 6, our accountable wiretapping architecture is highly scalable, and can easily handle a high rate of heartbeat messages, further diminishing the LOG's ability to infer linkage between encrypted events.

Court Audits. In a court audit, the court auditor verifies the Integrity, Completeness, and Date Compliance of the records returned by the LOG. Integrity is guaranteed through the use of digital signatures.

The court auditor verifies Completeness by searching for gaps in the wiretap records' sequence numbers (I_i). If the sequence numbers contains gaps, then the returned results are clearly incomplete. In the case that there are no gaps in the sequence numbers, M_α is a `start` message, and M_β is a `stop` message, then the returned records must be complete, since the `start` and `stop` messages "bookend" the wiretap, and the authenticity of all messages are verified via the ENCRYPTOR's digital signature. However, in all other cases, a corrupt LOG may omit records at the beginning or end of the queried time interval. Hence, sequence numbers by themselves are not sufficient to verify Completeness.

The use of heartbeat messages serves to detect such omissions. Let $M_\alpha, \ldots, M_\beta$ be the non-discarded (non-heartbeat) messages returned by the LOG, sorted in increasing order by the messages' timestamps (t_i). Additionally, let $[T_s, T_e]$ be the time interval specified by the court auditor, and t_α and t_β be the timestamps associated with M_α and M_β, respectively. Finally, let λ be the mean number

of heartbeat messages per minute produced by the Poisson process for the wiretap of interest. From the definition of a Poisson process, it follows that the probability that there is at least one heartbeat message between T_s and t_α is $1 - \frac{(\lambda \cdot (t_\alpha - T_s))^0 e^{-\lambda \cdot (t_\alpha - T_s)}}{0!} = 1 - e^{-\lambda \cdot (t_\alpha - T_s)}$, and $1 - e^{-\lambda \cdot (T_e - t_\beta)}$ for the time between t_β and T_e. Thus, the LOG's ability to omit records is constrained by λ. In summary, the court auditor can detect gaps by searching for noncontiguous sequence numbers, and can conclude with measurable confidence whether records were omitted at the endpoints by noting the presence or absence of heartbeats.

Finally, given the Completeness property, the court auditor can verify Date Compliance by issuing queries over various time intervals (in particular, $[-\infty, \infty]$) and reconciling the results with the wiretap creation and expiration dates specified by the Wiretap Authority.

Accounting Audits. Following the notation of Section 4.3, let M_a and M_z be the records returned by the LOG in an accounting audit. Additionally, let $\overline{\mathcal{B}}'$ be the purported summation of the aggregation blocks returned by the LOG. (Hence, if the LOG is honest and has captured all messages $M_a, M_{a+1}, \ldots, M_z$ from ENCRYPTOR, then $\overline{\mathcal{B}}' = \overline{\mathcal{B}}$.) In an accounting audit, the LOG must prove to the accountant that $\overline{\mathcal{B}}' = \overline{\mathcal{B}}$.

The authenticity and integrity of M_a and M_z can be easily checked by verifying their accompanying digital signatures. Using the argument described above, the use of the heartbeat message "bounds" omissions, and hence the ability of the LOG to omit many records before (resp. after) M_a (resp. M_z) is limited.

We define $\mathcal{R}'_A$ to be the second decrypted partition of $\overline{\mathcal{B}}'$ (the first nonce partition), and similarly, set $\mathcal{R}'_B$ to be the third decrypted partition of $\overline{\mathcal{B}}'$ (the second nonce partition). If $\overline{\mathcal{B}}' = \overline{\mathcal{B}}$, then $\mathcal{R}'_A = \sum_{k=a}^{z} R_k$ (by the definition of our protocol), and similarly, $\mathcal{R}'_B = \sum_{k=a}^{z} R_{k-1}$ (also by definition). Consequently, $\mathcal{R}'_A - \mathcal{R}'_B = R_z - R_{a-1}$.

The accountant can compute $\mathcal{R}'_A - \mathcal{R}'_B$ (from $\overline{\mathcal{B}}'$) and $R_z - R_{a-1}$ (from messages M_a and M_z that are also included in the LOG's response and whose digital signatures have previously been verified). If the LOG is honest and has captured all messages between t_a and t_z, then $\mathcal{R}_A - \mathcal{R}_B = R_z - R_{a-1}$.

An incorrect LOG or a LOG that has missed messages sent by the ENCRYPTOR can return a false result (i.e., $\overline{\mathcal{B}}' \neq \overline{\mathcal{B}}$). However, since the accountant knows R_z and R_{a-1} (from messages M_a and M_z), then to avoid detection, the LOG must return a $\overline{\mathcal{B}}'$ such that $\mathcal{R}'_A - \mathcal{R}'_B = R_z - R_{a-1}$. The LOG however does not have knowledge of R_{a-1} or R_z or $R_z - R_{a-1}$ (recall that the values are homomorphically encrypted and the LOG does not possess the decryption key).

The incorrect LOG may return a valid summation over an arbitrary subset of recorded messages – while not adhering to the time interval $[T_s, T_e]$. (In all other cases, the dispersion property of the cryptosystem ensures that the decryption of $\overline{\mathcal{B}}'$ appear random, since the LOG does not have the encryption key. Thus, the probability that $\overline{\mathcal{B}}'$ decrypts to the value of $R_z - R_{a-1}$ is negligible.) Since each nonce is chosen uniformly at random from a large space, the probability that the summations of random (and unknown) nonces $\mathcal{R}'_A$ and $\mathcal{R}'_B$ have the property $\mathcal{R}'_A - \mathcal{R}'_B = R_z - R_{a-1}$ is also negligible.

5.4. Operational Aspects

A limitation of our architecture is that it may be difficult to determine the *cause(s)* of a missed event. In particular, a missed message may be due to network loss, equipment failure, or the purposeful omission of a wiretap record. Court audits can reveal the *scope* of the omissions, which in turn may provide useful insights for statistical analysis (for example, to gauge the reliability of the network). However, in a lossy network, it is impossible to differentiate between messages that fail to reach their intended receivers due to network loss and those that are purposefully deleted in transit. Analyzing the network to detect potential causes of network loss (e.g., underprovisioning) and more closely scrutinizing the maintainer of the LOG are manual processes that will need to be undertaken by the courts. We note, however, that the courts currently lack any ability to detect gaps (as publicly highlighted by ex-Gov. Blagojevich's legal team [33]), regardless of their causes. The ability to detect missed events is a significant advancement, as it enables the court to assert with very high confidence whether or not a wiretap transcript is complete.

6. Evaluation

In this section, we describe our implementation of the ENCRYPTOR device and demonstrate its performance under realistic operating configurations and workloads.

6.1. Implementation

Given the lack of public information regarding existing CALEA wiretap implementations and the difficulty of procuring them, we chose to integrate our system with an Asterisk softswitch. Asterisk is an open source telephony program capable of bridging calls between the standard telephone network (PSTN) and voice-over-IP (VoIP) networks, including proprietary services such as Skype. Asterisk is capable of scripting telephony-related events, and in addition to a native scripting language, it allows event

Operation	1024 bit Agg. Block		2048 bit Agg. Block	
Hash Call Data	10.57μs	[10.54,10.59]	8.13μs	[8.12,8.15]
Enc. Agg. Block	31676μs	[31619,31733]	199310μs	[198951,199668]
Enc. Call Data	10.17μs	[10.15,10.19]	9.1μs	[9.08,9.12]
Sign Call Record	1193μs	[1191,1195]	1073μs	[1071,1075]
Transmission	1.8μs	[1.79,1.81]	0.57μs	[0.57,0.57]
Total Process	32756μs	[32697,32815]	200700μs	[200338,201061]

Table 2: Microbenchmarks of Call Record Event Generation with aggregate blocks using 1024 and 2048 bit moduli. Averages and 95% confidence intervals, shown in brackets, are based on 1000 runs.

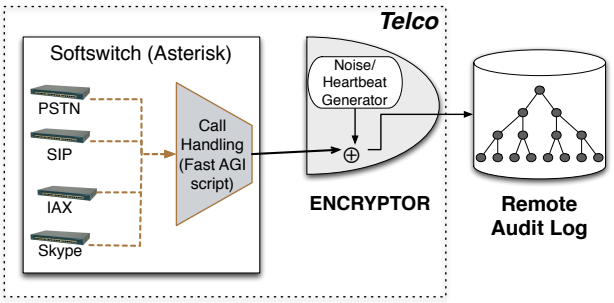

Figure 3: Our proof-of-concept ENCRYPTOR implementation. Within the telco facilities, the Asterisk softswitch handles call events and forwards them to the to the ENCRYPTOR, which communicates securely with a remote audit log.

flows to be passed off to other scripts and processes. One such service is the FastAGI server, which is generally used to allow a single switch to accomodate additional load by outsourcing the call handling responsibilities to additional machines. We run our FastAGI server locally to mark the barrier between a SWITCH (where the Delivery Function is implemented) and the ENCRYPTOR. We implemented the ENCRYPTOR as a Java process that checks calls against a list of wiretap orders provided by an Authority. Figure 3 provides an overview of our implementation, showing the flow of collected call data from the call handling script processed by Asterisk to the ENCRYPTOR and discrete events entered into the LOG, which can be accessed by auditors.

Wiretap Event Generation. All incoming Asterisk calls pass through the control of the ENCRYPTOR, which examines the call metadata to see if either communicant is subject to an Authority-issued wiretap. If this is the case, it begins to send encrypted call records to a remote LOG. Each record contains the message M_i that includes timestamp t_i, encrypted call data $E_r(\tau_i \mid\mid I_w)$, SHA1 checksum $h(\tau_i||I_w)$, and aggregation block $\mathcal{B}_i$; it also includes the signature $\sigma_{A^-}(M_i)$.

The aggregation block $\mathcal{B}_i$ is encrypted using the Paillier public-key cryptosystem [21]. Paillier possesses the additive homomorphic property that $(E(m_1) \cdot E(m_2)) \bmod n^2 = (m_1 \oplus m_2) \bmod n$, which we use to perform ciphertext addition in our accounting audit. The Paillier keys are generated to support a 1024-bit modulus and $1-2^{-64}$ prime certainty. This creates a field of 308 digits with which to encode aggregate information.

To protect the switch from traffic analysis, the ENCRYPTOR creates a stream of cover traffic events. These events are generated according to a Poisson distribution using a secure PRNG. The frequency of the cover traffic pattern is scaled such that legitimate events are effectively obscured. Once encrypted, these events are seemingly identical to legitimate wiretap records. Because all records are signed with the private authenticity key A^-, the LOG cannot distinguish between cover and legitimate events upon their arrival. This helps obfuscate the timing information that would otherwise leak information to a dishonest maintainer. Additionally, the cover traffic events double as our heartbeat messages to the LOG.

Auditing. Our test implementation supports the two forms of audits described in Section 4.3. For *court audits*, a court auditor is able to issue a request for all records in the interval $[T_s, T_e]$. The LOG returns these encrypted events to the auditor. The auditor can then attempt to decrypt the returned records using its known record keys (r). Since an auditor may have access to only a subset of keys r, the (in)ability to decrypt serves as an avenue for enforcing finer-grained access control on log events. While full audits would require the attempted decryption of all events for record keys r, this is an offline and infrequent cost that can be parallelized across many cores.

For the *accounting audit*, the accountant issues a set of three commands. The first request is that the aggregate blocks of records $M_f, \ldots, M_l$ be summed. The next two requests are for the records M_f and M_l. The auditor can then audit the LOG for the given range (as described in Section 4.3) with guaranteed correctness up to the most recent heartbeat record.

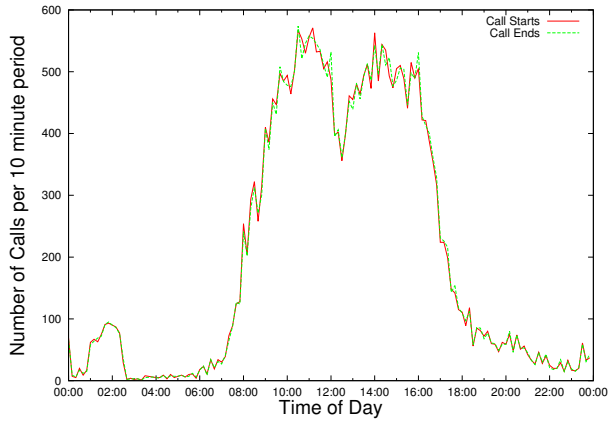

Figure 4: Profile of traffic of a major university from April 4th, 2011. Call count is grouped by ten minute increments.

6.2. Performance

In order to evaluate the throughput of our implementation, we first performed microbenchmarking tests on the different steps of creating an individual event. These tests were performed on an Intel Xeon 2.67 GHz quadcore processor; the machine had 8GB memory and was running Linux 2.6.35. We microbenchmarked the following steps: call data hashing, aggregate block encryption (using the Paillier public-key cryptosystem), symmetric encryption of actual call data (AES used in CBC mode), signing (1024-bit DSA signatures), and transmission over an open TCP socket, with the benchmarks executed within a single-threaded process. The results of these benchmarks are displayed in Table 2.

Performance degrades as the size of the aggregate block increases. Because of the overhead of encryption with the additive homomorphic property, this parameter has a significant impact on the ENCRYPTOR's throughput. Using a 1024-bit modulus, the aggregate block encryption step represents 96.4% of the cost of record generation. Event signing consumes 3.5% of generation, and the remaining steps (record initialization, symmetric encryption, transmission) contribute less than 1%. As benchmarked, our 1024-bit ENCRYPTOR implementation can generate 30.53 events per second.

For our proof-of-concept implementation, each record contains two 64-bit nonce sequence numbers which serve as unique identifiers for the record and its predecessor. Setting a large nonce size can all but eliminate the possibility of predicting the sequence number. The aggregate block size also imposes an upper bound on the maximum size of the log. Each field in the aggregate block can only accommodate the addition of a fixed number of events before the sum bleeds into the next field of the block. Prior to this occurring, the log must be rotated. In our implementation, we designed the 1024 bit aggregate block to accomodate 2^{168} events before rotation was required.

At the remote LOG, events may be entered into it at roughly 39,000 events per second. The system throughput is therefore entirely dependent on the speed of the ENCRYPTOR. In a realistic scenario, many delivery functions at different central offices are sending traffic to a single location. Even with our unoptimized implementation, it is clear that our LOG is prepared to accommodate this one-to-many relationship.

Having determined our system throughput, we sought to test our implementation in real world conditions. To do this, we obtained an anonymized profile of all outgoing and incoming call data from a major university for a 24-hour period. No activity of any individual was exposed. The profile of this traffic is pictured in Figure 4. In our experiment, we initiated a SIP telephone call and generated a Call Start wiretap event for every call for the busiest call windows of the day. As the peak call count per 10 minute window was only 571 calls, our ENCRYPTOR implementation was able to easily handle this traffic. With similar ease, we could have logged Call End wiretap events along with other assorted CALEA event types. Over the course of the simulation, our ENCRYPTOR operated at less than 3.2% of its maximum throughput.

These results can also be extrapolated to national telephony traffic. In 2003, AT&T reported that it handled 3,472 calls per second on average [13]. By our benchmark numbers using the 1024-bit aggregate block, we could generate Call Start events for 0.87% of this traffic. Our actual, Asterisk-attached implementation is already multithreaded, so it is not unreasonable to imagine that we would be able to log wiretap events for 10% of national call traffic on a single multicore machine. We believe that these numbers far exceed the actual number of wiretap events (based on the evidence below), demonstrating that our system contains significant capacity to record attacks such as those presented by Sherr et al. [32].

Yet another method of evaluating our implementation is to compare it to published wiretap statistics. In 2008, there were 20,899 pen register orders and 386 full content intercept orders [11]. Because our current implementation does not support content interception, we will consider only the pen registers. We requested the specific call rates of the university's most called number (i.e., the main campus exchange), with 120 outgoing and 280 received calls in a single day. Let us assume that each wiretap target will initiate and receive this extremely high number of calls every day, for a total of 400 Call Start and Call End events. Conservatively assume that in 2008 there were 20,899 simultaneous pen registers on a given day. We would then expect roughly eight million events over the course of that day. At 30.53 events per second, our implementation can generate nearly

three million events per day, and could therefore accommodate all of the 2008 pen registers using only three machines.

6.3. Discussion

Our accountable wiretapping architecture is not intended to be a panacea for providing more robust telecommunication. Auditing is inherently a reactive security mechanism, capturing incorrect behavior only after it occurs. Insiders with physical access to service providers' equipment can exfiltrate information, and likely do so while evading our auditing infrastructure. In general, preventing unauthorized data exfiltration is a difficult and open problem – one which our techniques are not intended to solve. Rather, we present a wiretapping audit process and associated protocols that provide a first step towards an architecture that is more robust against manipulation and privacy violations. Although we do not prevent all plausible attacks against wiretap systems, we do ensure that wiretap data can be authenticated and that omissions in wiretap transcripts are detectable.

7. Related Work

Telephony systems and their users have long been subject to attack. The majority of such networks remain susceptible to eavesdropping attacks, due to traffic being unencrypted in the provider network cores [1] or protected by weak cryptographic algorithms over the air [14, 4, 3, 2, 23, 17]. Even the contents of VoIP traffic protected by strong cryptographic algorithms can be exposed by comparing packet size and interarrival times to language constructs [46, 45, 44]. Instances of these networks have also proven susceptible to a range of other attacks including the exploitation of in-band signaling [26], fraudulent billing [25] and overload [35, 36, 38, 37].

Legal interception laws mandate that telephony providers must provide law enforcement access to certain call content and metadata. While undetectable by the party or parties being monitored, wiretap systems are vulnerable to a range of attacks. Sherr et al. [31] demonstrated the ability to prevent call audio from being recorded by injecting in-band signaling tones into a conversation. Moreover, this work also demonstrated the ability to confuse dialed-digits logs, making the actual endpoint of a call difficult to discern for the eavesdropper. Follow-on work by these same authors [32] also demonstrated the ability to overload wiretapping-enabled switches, causing potentially critical data to be lost before an action could be logged. Such attacks are the direct inspiration for this work.

Reliable and tamper-evident logging are critical components in a range of other systems. A number of non-cryptographic schemes rely on the "write-once" nature of their associated storage medium (e.g., PROMs) [47, 22, 20]. While effective at preventing overwriting, such records are difficult to audit remotely. Others have instead suggested software-based solutions relying on a variety of schemes including forward-secure signature [29, 16], distributed timestamping [15, 7, 19, 27], Bloom Filters [30] and Merkle hash trees [10]. While these schemes provide strong guarantees, they are not sufficient in our application because they typically assume that the event generator and the auditor are the same party. Moreover, these techniques do not support secure aggregation of records over user-specified time periods. Because of the increased interest in third party collection and storage of wiretap records [43], a more robust solution is required.

8. Conclusion

While legal wiretaps can be a central part of a legal investigation, it is important to provide oversight to help limit abuse and ensure compliance. We have proposed the first distributed auditing system for existing CALEA-compliant wiretaps. In our system, ENCRYPTORs are added to each CALEA device, and send encrypted audit records to a LOG. The LOG is trusted only to store encrypted records, which it can provide in an court audit, and to compute homomorphically-encrypted audit statistics, which it can deliver during an accounting audit. This allows a court to examine all records for a particular wiretap as needed, and for inexpensive periodic accounting over all wiretaps by other judicial authorities. The LOG is unable to determine wiretap contents nor even what wiretaps exist, and any log alterations are easily detected.

We have shown that our system is resistant to target-initiated DoS attacks, can detect illicit wiretaps by employees of the phone company or wiretaps that exceed their legal lifetime, and will detect a dishonest LOG that attempts to alter wiretap records. Our implementation results demonstrate that the system will easily scale to a level sufficient to monitor all known US wiretaps on three commodity machines.

Acknowledgments

We thank the anonymous reviewers for their comments and suggestions. This work is partially supported by NSF Awards CNS-0916047, CNS-1064986, CNS-1118046, CNS-1117943, CNS-0964566, and CAREER CNS-0952959. The views expressed are those of the authors and do not reflect the official policy or position of the National Science Foundation or the U.S. Government.

References

[1] 3rd Generation Partnership Project. Technical Specification Group Services and System Aspects; 3G Security; Network Domain Security; MAP application layer security . Technical Report 3GPP TS 33.200 v7.0.0.

[2] E. Barkhan, E. Biham, and N. Keller. Instant Ciphertext-Only Cryptanalysis of GSM Encrypted Communication. In *Proceedings of the Annual International Cryptology Conference (CRYPTO)*, 2003.

[3] E. Biham and O. Dunkelman. Cryptanalysis of the A5/1 GSM Stream Cipher. In *Proceedings of INDOCRYPT*, 2000.

[4] A. Biryukov, A. Shamir, and D. Wagner. Real Time Cryptanalysis of A5/1 on a PC. In *Proceedings of the Fast Software Encryption Workshop*, 2000.

[5] M. Blaze and S. M. Bellovin. Inside RISKS: Tapping, Tapping on My Network Door. *Communications of the ACM*, 43(10), 2000.

[6] British Parliament. Regulation of Investigatory Powers Act 2000: Part IV: Scrutiny etc. of Investigatory Powers and of the Functions of the Intelligence Services, July 2000.

[7] A. Buldas, P. Laud, H. Lipmaa, and J. Willemson. Time-Stamping with Binary Linking Schemes. In *Proceedings of CRYPTO*, 1998.

[8] Cisco Systems, Inc. Cisco Voice Switch Services Configuration Guide for MGX Switches and Media Gateways Release 5.5.10, June 2009.

[9] Cisco Systems, Inc. Cisco BTS 10200 Softswitch Provisioning Guide, Release 5.0.x, May 2010.

[10] S. A. Crosby and D. S. Wallach. Efficient Data Structures for Tamper-evident Logging. In *USENIX Security Symposium (USENIX)*, 2009.

[11] Director of the Administrative Office of the United States Courts. Report of the Director of the Administrative Office of the United States Courts on Applications for Orders Authorizing or Approving the Interception of Wire, Oral, or Electronic Communications, April 2009. Covers 2008.

[12] Director of the Administrative Office of the United States Courts. Report of the Director of the Administrative Office of the United States Courts on Applications for Orders Authorizing or Approving the Interception of Wire, Oral, or Electronic Communications, June 2011. Covers 2010.

[13] K. Fisher and R. E. Gruber. PADS: Processing Arbitrary Data Streams. In *Proceedings of the Workshop on Management and Processing of Data Streams (DIMACS)*, 2003.

[14] J. D. Golic. Cryptanalysis of Alleged A5 Stream Cipher. In *Proceedings of EuroCrypt*, 1997.

[15] S. Haber and W. S. Stornetta. How to Timestamp a Digital Document. In *Proceedings of CRYPTO*, 1990.

[16] J. E. Holt. Logcrypt: Forward Security and Public Verification for Secure Audit Logs. In *Proceedings of the Australasian Workshops on Grid Computing and e-Research*, 2006.

[17] B. Krebs. Research May Hasten Death of Mobile Privacy Standard. `http://blog.washingtonpost.com/securityfix/2008/02/research_may_spell_end_of_mobi.html`, 2008.

[18] S. Landau. *Surveillance or Security?: The Risks Posed by New Wiretapping Technologies*. MIT Press, 2011.

[19] P. Maniatis and M. Baker. Secure History Preservation Through Timeline Entanglement. In *Proceedings of the USENIX Security Symposium (SECURITY)*, 2002.

[20] S. Mitra, W. W. Hsu, and M. Winslett. Trustworthy Keyword Search for Regulatory-compliant Records Retention. In *Proceedings of the International Conference on Very Large Data Bases (VLDB)*, 2006.

[21] P. Paillier. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *Annual International Cryptology Conference (CRYPTO)*, 1999.

[22] K. Pavlou and R. T. Snodgrass. Forensic Analysis of Database Tampering. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2006.

[23] S. Petrovic and A. Fuster-Sabater. An Improved Cryptanalysis of the A5/2 Algorithm for Mobile Communications. In *Proceedings of Communication Systems and Networks*, 2002.

[24] V. Prevelakis and D. Spinellis. The Athens Affair. *IEEE Spectrum*, 44(7):18–25, 2007.

[25] A. Ramirez. Theft Through Cellular 'Clone' Calls. *The New York Times*, April 7, 1992.

[26] R. Rosenbaum. Secrets of the Little Blue Box. *Esquire Magazine*, pages 117–125 and 222–226, October 1971.

[27] D. Sandler and D. S. Wallach. Casting Votes in the Auditorium. In *Proceedings of the USENIX/ACCURATE Electronic Voting Technology Workshop (EVT)*, 2007.

[28] C. Savage. U.S. Tries to Make It Easier to Wiretap the Internet. The New York Times, September 27 2010.

[29] B. Schneier and J. Kelsey. Secure Audit Logs to Support Computer Forensics. *ACM Transactions on Information and System Security (TISSEC)*, 1(3), 1999.

[30] K. Shanmugasundaram, H. Bronnimann, and N. Memon. Payload Attribution via Hierarchical Bloom Filters. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2004.

[31] M. Sherr, E. Cronin, S. Clark, and M. Blaze. Signaling Vulnerabilities in Wiretapping Systems. *IEEE Security & Privacy*, 3(6):13–25, November 2005.

[32] M. Sherr, G. Shah, E. Cronin, S. Clark, and M. Blaze. Can They Hear Me Now?: A Security Analysis of Law Enforcement Wiretaps. In *ACM Conference on Computer and Communications Security (CCS)*, 2009.

[33] M. Tarm. Rod Blagojevich Seeks to Toss Wiretaps. The Christian Science Monitor, February 22 2011.

[34] Telecommunications Industry Association (TIA). Lawfully Authorized Electronic Surveillance (J-STD-025-B). J-STD-025B, 2003.

[35] P. Traynor, W. Enck, P. McDaniel, and T. La Porta. Exploiting Open Functionality in SMS-Capable Cellular Networks. *Journal of Computer Security (JCS)*, 16(6):713–742, 2008.

[36] P. Traynor, W. Enck, P. McDaniel, and T. La Porta. Mitigating Attacks On Open Functionality in SMS-Capable Cellular Networks. *IEEE/ACM Transactions on Networking (TON)*, 17(1), 2009.

[37] P. Traynor, M. Lin, M. Ongtang, V. Rao, T. Jaeger, T. La Porta, and P. McDaniel. On Cellular Botnets: Measuring the Impact of Malicious Devices on a Cellular Network Core. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2009.

[38] P. Traynor, P. McDaniel, and T. La Porta. On Attack Causality in Internet-Connected Cellular Networks. In *Proceedings of the USENIX Security Symposium (SECURITY)*, 2007.

[39] Communications Assistance for Law Enforcement Act. Pub. L. No. 103-414, 108 Stat. 4279, 1994. (codified as amended in sections of 18 U.S.C. and 47 U.S.C. Sect. 229, 1001-1010, 1021).

[40] United States Congress. Pub. L. No. 106-197 amended USC §2519(2)(b), USA.

[41] United States Congress. Omnibus Crime Control and Safe Streets Act of 1968: Title III. Pub. L. No. 90-351, 82 Stat. 197, USA, 1968. (codified as amended in 18 U.S.C. Sect. 2510-2522).

[42] U.S. Justice Department. Report on the Use of Pen Registers and Trap and Trace Devices by the Law Enforcement Agencies/Offices of the Department of Justice for Calendar Year 2008, 2008.

[43] Verint. STAR-GATE Comprehensive Service Provider Compliance with Lawful Interception and Data Retention Mandates, October 2007. Retrieved from `http://verint.com/communications_interception/file.cfm?id=51` on April 30, 2011.

[44] A. M. White, K. Snow, A. Matthews, and F. Monrose. Phonotactic Reconstruction of Encrypted VoIP Conversations: Hookt on fon-iks. In *IEEE Symposium on Security and Privacy (Oakland)*, 2011.

[45] C. Wright, L. Ballard, S. Coull, F. Monrose, and G. Masson. Spot me if you can: Uncovering spoken phrases in encrypted VoIP conversations. In *Proceedings of IEEE Symposium on Security and Privacy (OAKLAND)*, 2008.

[46] C. Wright, L. Ballard, F. Monrose, and G. Masson. Language Identification of Encrypted VoIP Traffic: Alejandra y Roberto or Alice and Bob? In *Proceedings of the USENIX Security Symposium*, 2007.

[47] Q. Zhu and W. W. Hsu. Fossilized Index: The Linchpin of Trustworthy Non-alterable Electronic Records. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2005.