Light Sensor Based Covert Channels on Mobile Devices

Mila Dalla Preda[a], Claudia Greco[b], Michele Ianni[b], Francesco Lupia[b], Andrea Pugliese[b]

[a]*Department of Computer Science, University of Verona, Italy*
[b]*Department of Computer Engineering, Modeling, Electronics, and Systems, University of Calabria, Italy*

Abstract

The widespread adoption of light sensors in mobile devices has enabled functionalities that range from automatic brightness control to environmental monitoring. However, these sensors also present significant security and privacy risks within the Android ecosystem due to unrestricted access permissions. This paper explores how light sensor data can be used for covert communication through a novel, light-based out-of-band channel. We develop two approaches–Baseline and ResetBased–that use luminance values to encode and decode data. These methods tackle challenges that arise from data variability and the unpredictability of sensor event timings. To enhance data transmission accuracy, our methods employ a novel strategy for selecting luminance reference sequences and leverage mean-squared-error-based distance for decoding. Experimental results validate the effectiveness of our approaches and their potential for real-world applications.

Keywords: Covert channels, Light sensors, Mobile devices

1. Introduction

A huge number of mobile devices, including the vast majority of smartphones, are equipped with rather precise sensors that measure the intensity of

Email addresses: `mila.dallapreda@univr.it` (Mila Dalla Preda), `claudia.greco@unical.it` (Claudia Greco), `michele.ianni@unical.it` (Michele Ianni), `francesco.lupia@unical.it` (Francesco Lupia), `andrea.pugliese@unical.it` (Andrea Pugliese)

light in the surrounding environment. The adoption of light sensors enables various functionalities, such as (i) automatic brightness control, i.e., adjustment of screen brightness based on ambient light conditions, (ii) optimization of camera exposure and white balance, to help capture higher-quality images and videos, (iii) proximity sensing, to provide features such as automatic screen dimming during phone calls or gesture-based interactions, and (iv) environmental monitoring, such as indoor lighting control or outdoor illumination detection for urban planning purposes. However, in the Android ecosystem, the security and privacy implications of sensor data access are a significant concern. Sensors on Android devices provide a wealth of information that can be leveraged for a variety of legitimate purposes, on the other hand, the same data can be exploited for malicious purposes, such as eavesdropping on user inputs and inferring personal information. To complicate matters, certain sensors, *including the light sensor*, are exempted from the Android permission model, allowing *unrestricted access by any app installed on the device*. This opens an avenue for the exploitation of light sensor-based channels to transmit data to targeted devices.

In this paper, by exploiting light sensor data acquisition and its potential for covert communication, we develop a novel technique to create a light-based out-of-band Covert Channel (CC). The first important challenge we must face is related to the way the Android Sensor Framework provides data from the light sensor to apps on non-rooted devices. The framework requires registering a listener to monitor changes in luminance values, data are not sampled at specific instants but rather obtained when there is a significant change. This implies that if there are frequent changes in ambient light, the sensor may produce more events within a given period of time, whereas if the ambient light remains relatively stable, the sensor may produce fewer events. The rate at which events are received by the listener also depends on the magnitude of changes in luminance.

Another challenge arises because sensor values can fluctuate due to environmental factors or sensor noise. To counteract this, the sensor framework uses averaging techniques to smooth sudden changes and prevent problems such as erratic screen brightness adjustments. This averaging must be appropriately taken into account during data transmission.

The approaches we develop in this paper, named Baseline and ResetBased, encode and decode bytes using luminance values. Specifically, we employ reference sequences of luminance values and, to decode a sequence of values read by the light sensor, we employ a mean-squared-error-based distance.

The ResetBased approach is specifically designed to overcome the limitations of the Baseline approach, which suffers more from the unpredictability of light change events timing and value averaging.

Transmitting data by modulating luminance values poses another important challenge, regarding the selection of a specific set of reference luminance levels to use in both the encoding and decoding phases. If the overall range of available values is sufficiently wide, it can be split into multiple subranges, allowing for higher bitrates, this in turn depends on the resolution of luminance values that we can discern. However, even when using shades of grey as input to the light sensor, if we evenly split the overall range of shades into equal subranges, we do not obtain equally spaced subranges in the luminance space. A valuable property of our proposed approaches is that they are not restricted to the use of grey shades: we successfully tested inputs with varying colors that yielded the same luminance readings from the sensor as the original grey inputs.

The experimental assessment we carried out to evaluate and compare the performance of the Baseline and ResetBased approaches provided very interesting results and confirmed the feasibility of a light-based out-of-band CC. Hopefully, our study will contribute to the development of effective security countermeasures in the mobile ecosystem.

The remainder of the paper is organized as follows. In Section 2 we discuss existing literature papers that have dealt with communication established via out-of-band CCs, based on light or other transmission mediums. In Section 3 we provide background information and explain the main motivations of this work. In Section 4 we develop our proposed methodology for data transmission through light sensors. In Section 5 we describe the experimental assessment we carried out to validate our methodology. In Section 6 we discuss significant use cases. Finally, in Section 7 we draw our conclusions.

2. Related Work

Several studies have focused on the exploitation of light sensor-based channels to activate preexisting malware on targeted devices. Subramanian et al. [15] examined the use of a visible light channel to transmit a Trojan to a target device. Their experimental setup operated under moderate ambient lighting conditions, employing Morse code encoding and decoding, respectively, implemented on a transmitter (an HTC Inspire smartphone) and a receiver (an MTS310CB sensor node). The authors also discussed the potential of using a secret trigger mechanism for event-triggered attacks, allowing the activation of already installed trojans or malicious code in the compromised node via side channels. However, such a discussion is not supported by practical experiments specifically conducted with the light sensor. Uluagac et al. [16] examined the potential threats posed to the sensory channels of the Cyber-Physical Systems devices. They investigated the exploitation of vulnerable services on the iRobot Create robotic platform through its infrared light channel. By manipulating timer registers, the authors devised simple vulnerable services and initiated them via the infrared channel. Several other studies have demonstrated the exploitation of various sensors for diverse CC attacks. For example, smartphone accelerometers [1, 9], tire pressure monitoring system sensors in automobiles [3, 13], analog sensors vulnerable to signal injection attacks [6], and inaudible audio [4, 11, 12] have been used for covert communication or data transmission.

To the best of our knowledge, Hasan et al.'s work [5] represents the sole real effort within the scientific literature to address the exact same problem that we consider in this paper. The authors analyzed the use of command and control channels to trigger malicious behaviors within infected devices. Their study involved the testing of the light sensor on an Android platform, along with three additional sensors—microphone, magnetometer, and accelerometer—to demonstrate the feasibility of mobile malware activation through CCs. Experimentation with the light sensor involved varying parameters such as overhead lighting, screen sizes, distances, and angles, resulting in diverse data rates, peaking at 1 bit per second. However, several issues limit the feasibility and effectiveness of such a communication strategy. First, some inherent constraints imposed by the Android sensor framework must be addressed, such as the reliance on sensor event listeners, which prevents real-time access to sensor data, imposing the need for asynchronous event-driven data acquisition (see Section 3.4). As a consequence, the ability to precisely control the timing and duration of light emission for communication purposes is hindered. Moreover, the paper does not appropriately take into account the impact of sensor data averaging. This makes the task of detecting subtle variations in light intensity significantly harder, making communication techniques reliant on precise luminance levels or simple thresholding ineffective. Our research addresses these limitations by developing innovative strategies and implementing advanced ad hoc techniques to achieve a higher degree of robustness and reliability in our covert communication framework.

Other works explore the use of mobile sensors and the associated security and privacy implications. For instance, [8] examines the use of vibrations generated by mobile device sensors for covert data transmission. The modulation of visual vibrations on screens as a method to create hidden channels has been studied in [10]. In addition, the use of sensor data for user profiling and the deduction of personal information has been investigated [14]. Although this work focuses on a different aspect, it shares our concern about the privacy risks associated with the unregulated use of mobile sensors. These works, to which our paper aligns, share a common focus on the innovative use of mobile sensors and the associated risks of unrestricted access.

3. Motivation and Background

In this section, we provide a comprehensive overview of the fundamental concepts and technical aspects relevant to our methodology. We begin by discussing the fundamental principles of out-of-band communication, highlighting its importance in the realm of information security and covert communication techniques. We then explore the inner workings of light sensors in smartphones, discussing their functionality and their role in measuring ambient light. We also discuss the permissions model in Android, explaining the mechanisms by which apps gain access to various hardware components and sensitive data, and highlighting the implications of the current permissions framework for user privacy and data security. Finally, we highlight the complexities of data acquisition from light sensors, emphasizing the inherent limitations such as data averaging, which present challenges for utilizing light sensor data as a reliable covert communication channel.

3.1. Out-of-Band Covert Channels

Out-of-band CCs are hidden communication methods that operate beyond traditional data transmission pathways, often serving unauthorized or unintended purposes. They are CCs intentionally designed to be undetectable both for human perception and technical surveillance. According to Lampson [7], a CC is a communication channel that is not intended for information transfer at all. CCs differ from other categories of information hiding, such as steganography, anonymity, and subliminal channels: while steganography hides messages within data, anonymity conceals participant identities, and subliminal channels embed messages within communication to influence behavior, CCs aim to establish hidden communication paths within existing systems. Generally, CCs rely on existing resources within systems, networks, or protocols to establish hidden communication links, and the entities implementing a CC often exploit pathways not originally meant for communication.

Unlike other types of CC, such as single-host and network CCs, out-of-band CCs use alternative paths separate from primary communication channels [2]. Instead of operating within established communication infrastructures, out-of-band channels make use of resources or mechanisms not originally designed for communication purposes.

Specifically, out-of-band CCs leverage everyday devices and sensors, utilizing common hardware. With the extensive range of sensors and devices integrated into everyday hardware, there exist numerous feasible options for establishing out-of-band CCs without the need for hardware modification. These channels can take various forms, including acoustic, light-based, vibration-induced, magnetic, temperature-based, radio frequency, and others. Single-host CCs leverage shared resources, such as files, locks, or buses, or utilize resources available on a single host. In contrast, out-of-band CCs utilize a shared physical medium, including light, air, radio waves, and more.

Another key difference lies in their operational context. Single-host CCs operate within a shared operating system and hardware environment, whereas out-of-band CCs operate between isolated processes often running on physically separate devices. These isolated processes typically do not share resources except the shared medium. Out-of-band CCs also resemble network CCs in establishing hidden connections between processes across different hosts. However, while network CCs hide communication within established host links, out-of-band CCs create new, hidden connections without relying on such links. This implies that the two communicating processes do not necessarily have to be running on the hosts that constitute the endpoints of the communication, but they can be part of the communication path created for transferring information between a transmitter and a receiver. In contrast, with out-of-band CCs, communication is established in the absence of a traditional communication link between the two communicating parties and is created to generate a hidden general-purpose connection.

Typically, out-of-band CCs do not provide the high bandwidth capabilities found in conventional radio frequency channels. This arises from the fact that out-of-band channels are not designed as traditional communication pathways, such as those utilized in standard wireless or wired networks. However, despite their limitations in bandwidth, out-of-band CCs demonstrate remarkable efficiency in transferring information, albeit at relatively low bitrates.

The primary advantage of out-of-band CCs lies in their ability to bypass detection by security mechanisms and monitoring systems: their hidden nature makes them particularly attractive to malicious actors seeking to conduct covert activities without detection. For instance, cybercriminals may exploit out-of-band CCs to exfiltrate sensitive data from compromised systems without triggering security alarms. By leveraging alternative communication pathways that evade traditional detection methods, such as network traffic monitoring or intrusion detection systems, attackers can surreptitiously transfer stolen information to external servers or command-and-control infrastructure.

3.2. Light Sensors in Smartphones

The light sensors on smartphones play a crucial role in capturing ambient light intensity, enabling various functionalities and enhancing the user experience. They operate based on the principles of photodetection, converting light energy into electrical signals, and typically consist of a photodetector, such as a photodiode or phototransistor, which generates a current proportional to the intensity of incident light. The output signal is then processed by the device's hardware and software to provide relevant information about ambient light conditions.

Such sensors serve multiple purposes, including (i) automatic brightness control, i.e., adjustment of screen brightness based on ambient light conditions, (ii) camera enhancements, i.e., by optimizing exposure settings and white balance adjustment, they help capture high-quality images and videos under varying lighting conditions, (iii) proximity sensing, when integrated with proximity sensors, to enable features such as automatic screen dimming during phone calls or gesture-based interactions, and (iv) environmental monitoring, such as measuring ambient light levels for indoor lighting control or assessing outdoor illumination for urban planning purposes.

There are several types of light sensors commonly found in smartphones, each with its characteristics, applications, and trade-offs. *Photodiodes* are semiconductor devices that generate a current proportional to incident light intensity. They are widely used in smartphones because of their high sensitivity and fast response times. However, they may suffer from limited spectral sensitivity and susceptibility to noise. *Phototransistors* are light-sensitive transistors that amplify the current produced by incident light. They offer higher sensitivity and lower noise compared to photodiodes, making them suitable for applications requiring more precise light detection. Finally, *Complementary Metal-Oxide-Semiconductor (CMOS) image sensors* serve dual functions as light sensors and imaging devices. They capture light and convert it into digital images, enabling advanced camera features such as high dynamic range imaging and low-light performance enhancement. CMOS image sensors offer excellent image quality and versatility but may consume more power and require complex processing algorithms.

Smartphone manufacturers consider factors such as sensitivity, response time, power consumption, integration complexity, and low-light performance. Table 1 compares the features of different types of light sensors.

	Photodiodes	Phototransistors	CMOS image sensors
Sensitivity	Worse	Better	Worse
Response time	Better	Worse	Worse
Power consumption	Better	Better	Worse
Integration complexity	Better	Better	Worse
Low-light performance	Worse	Worse	Better

Table 1: Types of light sensors and their features.

3.3. Android Permissions Model

The Android operating system implements a permission-based model to protect both user privacy and system integrity. This model requires applications to declare their intention to access sensitive data or system features in their *manifest* file. Subsequently, the system prompts the user for explicit approval before granting the requested permissions.

Over the years, this model has evolved to offer finer granularity and more control to users, especially with the introduction of runtime permissions in Android 6.0 (API level 23).[1] Despite these advancements, the security and privacy implications of sensor data access via permissions remain a significant concern. Sensors on Android devices provide a wealth of information that can be leveraged for a variety of legitimate purposes, ranging from enhancing user

[1] `https://developer.android.com/about/versions/marshmallow/android-6.0-changes`

experiences with context-aware applications to enabling health monitoring and fitness tracking. However, the same data can be exploited for malicious purposes, such as eavesdropping on user inputs (through motion sensors) and inferring personal information (via location and activity recognition sensors).

To mitigate such risks, Android categorizes sensors and their corresponding data into different protection levels, requiring apps to obtain specific permissions to access them. For instance, accessing GPS location data needs the *ACCESS_FINE_LOCATION* permission, while reading health-related sensors like the heart rate monitor requires the *BODY_SENSORS* permission.

Despite these protective measures, certain sensors are exempt from the permission model, allowing unrestricted access by any app installed on the device. *One such sensor is the light sensor*, which has traditionally been considered non-sensitive, as it was utilized primarily for adjusting screen brightness to accommodate varying lighting conditions. Apps can access data from the light sensor without any permission.

The substantial security limitation posed by the absence of dedicated permission for accessing light sensor data allows us to develop our communication scenario.

3.4. Android Sensor Data Acquisition

The acquisition of light sensor values in Android typically involves registering a listener to monitor changes in luminance values. This asynchronous approach ensures that the data is not sampled at specific instants but rather obtained when there is a change in the ambient light level. The Android Sensor Framework facilitates this process by providing APIs for interacting with various sensors, including light sensors.

The following code snippet demonstrates how to acquire light sensor values in an Android application.

```
SensorManager sensorManager = (SensorManager)
                        getSystemService(Context.SENSOR_SERVICE);
Sensor lightSensor = sensorManager.getDefaultSensor(Sensor.TYPE_LIGHT);
SensorEventListener lightSensorListener = new SensorEventListener() {
    @Override
    public void onSensorChanged(SensorEvent event) {
        float luminance = event.values[0];
        // Process the luminance value
    }
    @Override
    public void onAccuracyChanged(Sensor sensor, int accuracy) {
        // Handle accuracy changes if necessary
    }
};
```

```
sensorManager.registerListener(lightSensorListener, lightSensor,
                               SensorManager.SENSOR_DELAY_NORMAL);
```

In this example, a `SensorManager` instance is obtained, and the default light sensor is retrieved. A `SensorEventListener` is then defined to handle changes in light intensity. The listener is registered with the `SensorManager`, specifying the delay in data delivery.

It is important to underline that when a light sensor listener is registered with the Android Sensor Framework, it starts listening for changes. The framework continuously monitors the sensor's output and notifies the listener whenever there is a change in the level that exceeds a certain threshold. This means that if there are frequent changes in the ambient light intensity, such as variations due to changes in lighting conditions or movement of the device in different environments, the sensor may produce more events within a given period of time. In contrast, if ambient light remains relatively stable, the sensor may produce fewer events as there are fewer changes to report. Therefore, the rate at which events are received by the listener depends on the frequency and magnitude of changes, reflecting real-world variations in lighting conditions.

Another important aspect to consider is that the values obtained from light sensors may exhibit fluctuations due to environmental factors or sensor noise. To mitigate the impact of these fluctuations, it is common practice to apply averaging techniques. Android provides mechanisms for averaging sensor values over a specified time span. By smoothing out abrupt changes, averaged values offer a more stable representation of ambient light intensity. For instance, when data from light sensors is used to adjust screen brightness dynamically based on ambient light conditions, instantaneous changes could lead to erratic screen brightness adjustments, particularly in scenarios where sudden changes occur, such as a brief reflection of sunlight. To mitigate such abrupt adjustments, averaging readings over a short period of time becomes crucial. By averaging the readings, sudden spikes or dips in ambient light intensity are smoothed out, resulting in more gradual and appropriate adjustments to screen brightness.

On rooted devices, it is possible to directly access the light sensor values from system files: they can be read from `/sys/class/sensors/light_sensor/lux` or `/sys/class/sensors/light_sensor/raw_data`, without relying on the Android Sensor Framework. Although rooted devices offer the capability to directly access light sensor values from system files, it is important to note that they do not necessarily represent a realistic threat model for a broader user base, since the vast majority of Android users do not have any necessity for rooting their devices. Therefore, our work takes into account the complexities inherent in non-rooted devices. Since they only allow the use of the Android Sensor Framework for accessing light sensor values, they introduce challenges such as the inability to access instantaneous data and the use of averaged values, which must be addressed to effectively utilize light sensor data for covert communication. In our proposal, we face these challenges, providing insights into how our approach mitigates these limitations and offers robust solutions for leveraging light sensors as a covert communication mean on any Android device (both rooted and non-rooted).

4. Methodology

In this section, we describe our proposed methodology. After giving preliminary definitions, we develop a first approach, named Baseline, to encode and decode bytes using luminance values. Specifically, we employ a "reference sequence" for each possible byte, and to decode a sequence of luminance values, we minimize a mean-squared-error-based distance between the sequence itself and the reference sequences. Then, we discuss and formalize a more advanced approach, named ResetBased, that is designed to overcome the limitations of the Baseline approach.

4.1. Preliminaries

We assume the existence of an ordered set $\mathcal{T}$ of time points,[2] an ordered set $\mathcal{V}$ of luminance values, and an ordered set $\mathcal{L} \subseteq \mathcal{V}$ of reference luminance levels. We also write $\mathcal{B}$ to represent the set of all possible bytes.

We start by formalizing a sequence of luminance values.

Definition 1 (Sequence of Luminance Values). *A* sequence of luminance values *is a set* $S = \{(ts_1, v_1), (ts_2, v_2), \ldots, (ts_n, v_n)\}$ *where,* $\forall i \in [1, n]$,

- $ts_i \in \mathcal{T}$ *and, if* $i > 1$*, then* $ts_{i-1} < ts_i$*;*
- $v_i \in \mathcal{V}$ *represents the luminance value at time* ts_i*.*

[2]In the remainder of the paper, we assume that time points are measured in milliseconds.

We denote the duration *of S as $duration(S) = ts_n - ts_1$. We say that a sequence is* zero-aligned *iff $ts_1 = 0$.*

It should be observed that, given a sequence $S = \{(ts_1, v_1), (ts_2, v_2), \ldots, (ts_n, v_n)\}$, we can always derive its zero-aligned counterpart as $\{(ts'_1, v_1), (ts'_2, v_2), \ldots, (ts'_n, v_n)\}$ where $\forall i \in [1, n]$, $ts'_i = ts_i - ts_1$. *From now on, we assume w.l.o.g. that all sequences of luminance values are zero-aligned.*

Sequences can be projected on time ranges. Given a sequence $S = \{(ts_1, v_1), (ts_2, v_2), \ldots, (ts_n, v_n)\}$ and two timestamps ts', ts'' with $ts_1 \leq ts' < ts'' \leq ts_n$, the *subsequence of S over the range $[ts', ts'']$* is $S_{[ts', ts'']} = \{(ts, v) \mid (ts, v) \in S, ts' \leq ts \leq ts''\}$. *We assume w.l.o.g. that subsequences of luminance values are zero-aligned as well.* This can be easily obtained by considering the zero-aligned counterparts of the subsequences.

The sequence of luminance values can also be concatenated by appropriately adjusting their time points, as shown in Definition 2.

Definition 2 (Sequence Concatenation). *Given two sequences $S = \{(ts_1, v_1), (ts_2, v_2), \ldots, (ts_n, v_n)\}$ and $S' = \{(ts'_1, v'_1), (ts'_2, v'_2), \ldots, (ts'_m, v'_m)\}$, their concatenation $S \mid S'$ is the sequence $\{(ts_1, v_1), (ts_2, v_2), \ldots, (ts_n, v_n), (ts_n + ts'_1 + 1, v'_1), (ts_n + ts'_2 + 1, v'_2), \ldots, (ts_n + ts'_m + 1, v'_m)\}$.*

Observe that, by definition, $S \mid \emptyset = \emptyset \mid S = S$. We also formalize how to derive functions from sequences and *vice-versa*.

Definition 3 (Sequence-Derived Function). *Given a sequence $S = \{(ts_1, v_1), (ts_2, v_2), \ldots, (ts_n, v_n)\}$, the* function derived from S *is a mapping $f_S : [ts_1, ts_n] \to \mathcal{V}$ defined as*

$$f_S(ts) = v_i + (ts - ts_i) \cdot \frac{v_{i+1} - v_i}{ts_{i+1} - ts_i}$$

with $ts_i \leq ts \leq ts_{i+1}$.

Definition 4 (Function-Derived Sequence). *Given a function $f : [0, ts'] \to \mathcal{V}$ with $ts' \in \mathcal{T}$, the* sequence derived from f *is a sequence $S_f = \{(ts, f(ts)) \mid ts \in [0, ts']\}$.*

In other words,

- f_S is defined over the full range $[ts_1, ts_n]$ and linearly interpolates the values between two consecutive pairs in S;
- S_f simply defines a pair for each time point in the domain of f.

We write $M(S)$ to denote the message encoded by sequence S, and $bits(M(S))$ to denote the number of bits of information in $M(S)$. As a consequence, the bitrate obtained through S is

$$\frac{bits(M(S))}{duration(S)}.$$

4.2. Baseline Approach

The Baseline approach is built upon the idea of associating one out of 4 different luminance values to each possible pair of bits. Then, a sequence of luminance values encodes a byte by simply concatenating the 4 luminance values corresponding to the 4 pairs of bits in the byte. In the decoding phase, we employ a "reference sequence" for each possible byte and, to decode a sequence S of luminance values, we choose the reference sequence that minimizes the distance with S.[3]

Formally, for each byte $B \in \mathcal{B}$, we assume the existence of a *reference sequence* S_B of luminance values, with $duration(S_B) = T$, and its corresponding sequence-derived function f_{S_B}.

Then, to decode an input sequence of luminance values S with $duration(S) = T$, we define the message encoded in S as

$$M(S) = \operatorname*{arg\,min}_{B \in \mathcal{B}} \sum_{ts \in [0,T]} \big(f_{S_B}(ts) - f_S(ts)\big)^2.$$

[3]We fixed the number of luminance values to 4 as a consequence of various considerations. First, after a preliminary empirical analysis, the use of 4 values appeared to provide the best compromise between data transmission speed and error mitigation. In addition, given the fact that – as shown in Section 5 – we obtain satisfying results with 4 values, using 2 values is guaranteed to work well from the point of view of error mitigation. Thus, the only expected consequence of using 2 values would be that of lowering bitrates. Finally, the results we obtained with the use of 4 values were already better than those provided by our only direct competitor (see Section 5). Using more levels would definitely give us the possibility of increasing bitrates, but would also increase the complexity of decoding, with potentially worse error rates. We thus plan to investigate this as future work.

As a consequence, under the Baseline approach, we can obtain a bitrate equal to $\frac{bits(M(S))}{duration(S)} = \frac{8}{T}$ bits/s.

We build reference sequences as follows. We fix $\mathcal{L} = \{\mathsf{B}, \mathsf{DG}, \mathsf{LG}, \mathsf{W}\}$ with $\mathsf{B} < \mathsf{DG} < \mathsf{LG} < \mathsf{W}$.[4] We associate each luminance level in $\mathcal{L}$ with a pair of bits: B/00, DG/01, LG/10, and W/11. Then, for each byte $B \in \mathcal{B}$:

1. We build an input to the light sensor that includes 4 luminance values, each corresponding to a pair of bits in B and lasting $T/4$ ms.
2. We perform N runs—for each run $r \in N$:
   (a) We record the sequence of luminance values S_r obtained in the run.
   (b) We derive function f_{S_r}.
3. We define the average function as $f_{avg}(ts) = \frac{\sum_{r \in N} f_{S_r}(ts)}{N}$.
4. We derive the reference sequence for B from f_{avg}, i.e., we compute $S_B = S_{f_{avg}}$.

Example 1. *Assume* $\mathsf{B} = 0$, $\mathsf{DG} = 100$, $\mathsf{LG} = 200$, $\mathsf{W} = 300$, *and* $T = 2,000$. *The inputs to the light sensor built in Step 1 of the process for bytes 00011011, 00111000, and 00111011 are shown in Figure 1.*

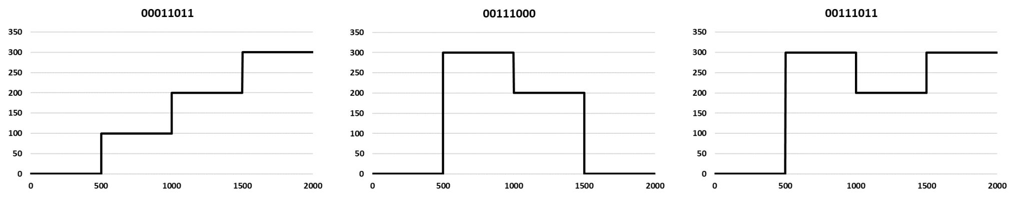


Figure 1: Inputs built for different bytes under the Baseline approach.

4.3. *ResetBased Approach*

As the experimental assessment will show (Section 5), the Baseline approach has a severe limitation. When transitioning from the darkest to the brightest luminance levels (and *vice-versa*), the averaging process in the sensor data acquisition brings along a risk of incomplete transitions, as the sensor readings may not reach the intended luminance values. This phenomenon is

[4]For now, we just use B, DG, LG, and W to represent the four different luminance levels employed. In Section 4.4, we will discuss how we choose specific shades of grey.

particularly pronounced in scenarios where rapid transitions occur or when environmental factors influence the sensor readings. As a result, there is a potential for ambiguity in the received data, with values falling within intermediate luminance levels being misinterpreted.

To address this, we developed the ResetBased approach that incorporates additional *reset values* into the communication protocol. These values are strategically inserted to facilitate smoother transitions between luminance levels, thereby ensuring that the intended luminance values are reached reliably. While reset values introduce a slight delay in the communication process (and obviously reduce bitrates), they play a crucial role in mitigating the impact of incomplete transitions caused by the averaging process.

Formally, assume we want to encode a sequence $\{B_1, \ldots, B_n\}$ of bytes. We fix $\mathcal{L} = \{\mathsf{B}, \mathsf{DG}, \mathsf{LG}, \mathsf{W}\}$ with $\mathsf{B} < \mathsf{DG} < \mathsf{LG} < \mathsf{W}$ and we associate each luminance level in $\mathcal{L}$ with a pair of bits: $\mathsf{B}/00$, $\mathsf{DG}/01$, $\mathsf{LG}/10$, and $\mathsf{W}/11$. Given a byte B, we denote the luminance level associated with the k-th pair of bits in B as $bitplev(B, k)$.

For each byte $B_i \in \{B_1, \ldots, B_n\}$, we build four sequences S_{ij} with $j \in [1, 4]$. Each S_{ij} contains a pair $(ts, bitplev(B_i, j))$ for each $ts \in [0, \frac{T}{2}]$. *It should be observed that, differently from the Baseline approach, for ease of presentation we now use T to refer to the duration of the encoding of four bits instead of a full byte.*

The full sequence $\{B_1, \ldots, B_n\}$ is encoded as the sequence of luminance values

$$S_{enc}(\{B_1, \ldots, B_n\}) =$$

$$S_{11} \mid R_{11/12} \mid S_{12} \mid R_{12/13} \mid S_{13} \mid R_{13/14} \mid S_{14} \mid R_{14/21} \mid$$

$$S_{21} \mid R_{21/22} \mid S_{22} \mid R_{22/23} \mid S_{23} \mid R_{23/24} \mid S_{24} \mid R_{24/31} \mid$$

$$\ldots$$

$$S_{n1} \mid R_{n1/n2} \mid S_{n2} \mid R_{n2/n3} \mid S_{n3} \mid R_{n3/n4} \mid S_{n4}$$

where $R_{ij/i'j'}$ are possibly empty *reset sequences* defined as

- $R_{ij/i'j'} = \emptyset$ if $|bitplev(B_i, j) - bitplev(B_{i'}, j')| \neq \mathsf{W} - \mathsf{B}$ (*no reset* case);
- $R_{ij/i'j'} = \{(0, bitplev(B_{i'}, j')), (\frac{T}{2}, bitplev(B_{i'}, j'))\}$ otherwise (*reset* case).

In order to evaluate the bitrate we can obtain under this encoding, let us denote the number of resets needed as $r \in [0, 4 \cdot n - 1]$. The duration of the sequence we build is then

$$duration(S_{enc}(\{B_1, \ldots, B_n\})) = n \cdot 2 \cdot T + r \cdot \frac{T}{2}.$$

We can therefore obtain a bitrate equal to

$$\frac{bits(M(S_{enc}(\{B_1, \ldots, B_n\})))}{duration(S_{enc}(\{B_1, \ldots, B_n\}))} = \frac{n \cdot 8}{n \cdot 2 \cdot T + r \cdot \frac{T}{2}}.$$

As a consequence:

- In the best case, $r = 0$ and the bitrate is $\frac{4}{T}$.
- In the average case, $r = \frac{4 \cdot n - 1}{8}$ and the bitrate is $\frac{n \cdot 8}{T \cdot (\frac{9}{4} \cdot n - \frac{1}{16})}$.
- In the worst case, $r = 4 \cdot n - 1$ and the bitrate is $\frac{n \cdot 8}{T \cdot (4 \cdot n - \frac{1}{2})}$.

Example 2. *Assume* $\mathsf{B} = 0$, $\mathsf{DG} = 100$, $\mathsf{LG} = 200$, $\mathsf{W} = 300$, *and that we want to encode a sequence* $\{B_1, B_2, B_3\}$ *containing the same 3 bytes of Example 1, i.e.,* $B_1 =$ *00011011,* $B_2 =$ *00111000, and* $B_3 =$ *00111011. Recall that, for any byte* B*,*

- $bitplev(B, k) = \mathsf{B}$ *iff the k-th pair of bits in B is 00;*
- $bitplev(B, k) = \mathsf{DG}$ *iff the k-th pair of bits in B is 01;*
- $bitplev(B, k) = \mathsf{LG}$ *iff the k-th pair of bits in B is 10;*
- $bitplev(B, k) = \mathsf{W}$ *iff the k-th pair of bits in B is 11.*

Table 2 shows the luminance levels associated with each pair of bits in the sequence.
The reset sequences are nonempty in the following reset cases:

- $R_{14/21} = \{(0, bitplev(B_2, 1) = \mathsf{B}), (\frac{T}{2}, bitplev(B_2, 1) = \mathsf{B})\}$*;*
- $R_{21/22} = \{(0, bitplev(B_2, 2) = \mathsf{W}), (\frac{T}{2}, bitplev(B_2, 2) = \mathsf{W})\}$*;*
- $R_{31/32} = \{(0, bitplev(B_3, 2) = \mathsf{W}), (\frac{T}{2}, bitplev(B_3, 2) = \mathsf{W})\}$.

	1-st pair	2-nd pair	3-rd pair	4-th pair
B_1	B	DG	LG	W
B_2	B	W	LG	B
B_3	B	W	LG	W

Table 2: Luminance levels associated with each pair of bits in the sequence $\{B_1, B_2, B_3\}$.

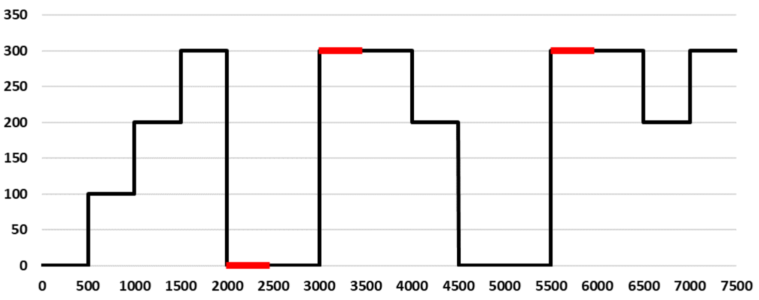


Figure 2: Example sequence of luminance values under the ResetBased approach.

Figure 2 depicts $S_{enc}(\{B_1, B_2, B_3\})$ when $T = 1,000$. In the figure, red segments correspond to reset sequences.

For the decoding phase, assume we are given a sequence S of readings with $duration(S) = q \cdot \frac{T}{2}$. The process is similar to the one defined for the Baseline approach, but we now have to consider reset sequences. As a consequence, we assume the existence of *a reference sequence for each combination of two pairs of bits*. The duration of a reference sequence is $\frac{3}{2} \cdot T$ if the two pairs are 00 11 or 11 00, and T otherwise. In our experiments, we build reference sequences as we did under the Baseline approach.[5]

Then, starting with an empty decoded sequence of bits, we apply the following steps:

1. We set $i = 0$.

[5]Under both the Baseline and ResetBased approaches, an initial synchronization phase is needed that basically consists in the trasmission of a predefined pattern before the actual encoded data. It should be observed that the sensor data gathered during this phase, specifically maximum and minimum luminance values, provide useful knowledge about environmental light conditions – this knowledge can be employed to normalize reference sequences and, as a consequence, to improve the similarity between reference sequences and sequences of sensor readings.

2. We look at subsequences $S_{[i \cdot \frac{T}{2}, (i+1) \cdot \frac{T}{2}]}$ and $S_{[i \cdot \frac{T}{2}, (i+2) \cdot \frac{T}{2}]}$ and determine the reference sequence that minimizes the distance with their sequence-derived functions.
3. We append the four bits corresponding to the reference sequence determined at Step 2 to the decoded sequence of bits.
4. If the four bits are 00 11 or 11 00, then we add 1 to i.
5. We add 1 to i. If $i < q$, then we go back to Step 2.

It should be observed that if the condition at Step 4 is satisfied, then the step "skips" a reset sequence.

4.4. Selection of Reference Luminance Levels

A major challenge in transmitting bits by modulating luminance values is selecting a specific set of reference luminance levels (set $\mathcal{L}$) for both encoding and decoding. This selection is crucial because it affects how much data we can transmit in a specific time frame. If the range of values is wide enough, we can divide it into several subranges to transmit more bits in a single frame. Naturally, the number of subranges we can use depends on the resolution of luminance values that we can discern. Although this resolution may vary slightly depending on the specific sensor employed, the primary factor influencing resolution is undoubtedly the ambient luminance. Given its potential variability, particular attention must be devoted to determining the breadth of range we can effectively utilize.

In our experiments, we determined the bounds of the range by capturing videos transitioning from a completely black screen to a completely white screen and then analyzing how changes in screen brightness influence the luminance readings obtained from the sensor. Of course, these values are subject to variation depending on several factors, including the sensor's orientation, the type of light source (e.g., incandescent bulb, neon lights, TV), the distance from the light source, and the brightness of the environment. If the light source is a monitor, other factors impact the luminance readings, such as the technology used (e.g., LCD, LED, OLED), the backlight, the size of the screen, and so on.

We partitioned the entire luminance range into four distinct levels, enabling the encoding of two bits per frame. As an illustrative example, we adopt grayscale colors: black, dark grey, light grey and white, each associated with a given luminance level. Several crucial considerations arise regarding the selection of the "perfect" shade for these colors.

First, the choice of the specific shades of grey cannot be arbitrary. To achieve a robust division of the luminance space, careful consideration is required. Although ideally, one might aim to evenly subdivide the range into four equal subranges, experimentation reveals that this kind of subdivision does not consistently yield equally spaced subranges in the luminance space. This discrepancy arises due to the imperfect alignment between the perceived luminance level by the sensor and the generated shades of grey in the video.

To tackle this, we created an alternative method to select optimal grey shades by producing a video that transitions from black to white and displays many grey shades for long periods. Concurrently, an Android application was developed to continuously monitor the luminance level while transitioning between black and white in the video. This application provided real-time feedback on the current luminance value, and a label in the video provided a visual representation of the corresponding shade of grey. By correlating the observed luminance values with the displayed shades of grey, equally spaced shades of grey were selected. This selection process facilitated the creation of four distinct subranges, each robustly embedding bits for data transmission.

Another important consideration is that our method is not restricted to the use of entirely grey frames. While this was convenient for experimental purposes, it is impractical and unrealistic in real-world scenarios to have videos composed solely of grey frames. We would like to emphasize that this constraint does not limit the applicability of our approach. In our experiments, we successfully generated images with varying colors that yielded the same luminance readings from the sensor as the original grey frames. The process for generating these values aligns with the methodology discussed earlier for selecting grey values.

Furthermore, to extend our technique to regular videos, we devised a strategy to modify frames from arbitrary videos. By adjusting parameters such as saturation, contrast, brightness level, and color space, we could alter the videos so that the resulting luminance could robustly encode our content. This approach allows for the utilization of diverse video content while maintaining effective data transmission. Moreover, for scenarios where direct control over a light source (e.g., a dimmable lightbulb) is possible, the process becomes even more straightforward, because the emitted light can be easily adjusted to embed the information to be transferred.

5. Experimental Assessment

In this section, we describe the experimental assessment we carried out in order to evaluate and compare the performance of the Baseline and ResetBased approaches.

The experiments were conducted using a 24-inch desktop LCD monitor in a room with natural light and overhead lights turned off, to maintain typical ambient lighting conditions. We varied the angle between the smartphone and the monitor from 90° to 0°, and the distance from 5cm to 40cm.

We fixed $T = 2,000$ for the Baseline approach and $T = 1,000$ for the ResetBased approach. Thus, the expected bitrates for a single transmitted byte were:

- $\frac{8}{2,000} = 4$ bits/s under the Baseline approach;
- $\frac{4}{1,000} = 4$ bits/s in the best case under the ResetBased approach;
- $\frac{8}{2,187.5} \simeq 3.66$ bits/s in the average case under the ResetBased approach;
- $\frac{8}{3,500} \simeq 2.29$ bits/s in the worst case under the ResetBased approach.

We recall that our only direct competitor reports bitrates peaking at 1 bit per second [5]. The expected bitrates with our proposed technique are between 2.99 and 4 times higher – the analysis we provide below confirms that these numbers are actually obtainable in practice.

We picked four bytes $B_1 = 00110011$, $B_2 = 00100010$, $B_3 = 11001100$, and $B_4 = 11011101$. We then built reference sequences for such bytes under each of the two approaches, with $N = 4$.

We ran 4 experiments for each pair of bytes and measured, under each approach, the average distance $avgdst(B_i, S_{B_j})$ between the encoding of B_i and the reference sequence of B_j (distances were computed, as defined in Section 4.2, on the corresponding sequence-derived functions).

Table 3 reports the values we obtained. The results show that, in all cases, the average distance between the encoding of B_i and the reference sequence of B_i itself is lower under the ResetBased approach—the decrease in the average distance was 86.74% for B_1, 36.83% for B_2, 21.07% for B_3, and 18.41% for B_4.

To better appreciate the more robust performance of the ResetBased approach, we defined the *uncertainty ratio* of a pair (B_i, B_j) of bytes with $i \neq j$

		S_{B_1}	S_{B_2}	S_{B_3}	S_{B_4}
Baseline	B_1	**923.06**	1,442.69	3,481.16	3,637.40
	B_2	1,635.02	**118.15**	1,363.47	4,983.92
	B_3	2,077.64	1,224.95	**146.56**	4,449.81
	B_4	1,905.44	4,531.43	4,816.03	**128.18**
ResetBased	B_1	**122.43**	4,559.19	12,576.56	3,647.77
	B_2	4,410.50	**74.64**	3,299.14	5,353.19
	B_3	13,678.89	3,837.80	**101.18**	7,916.14
	B_4	3,970.64	5,213.25	6,771.32	**104.59**

Table 3: Average distances (B_1 = 00110011, B_2 = 00100010, B_3 = 11001100, B_4 = 11011101).

as $\frac{avgdst(B_i,S_{B_i})}{avgdst(B_i,S_{B_j})}$. In other words, the higher this ratio, the more similar are (i) the average distance between the encoding of B_i and the reference sequence of B_i itself and (ii) the average distance between the encoding of B_i and the reference sequence of B_j. Obviously, higher uncertainty ratios (i.e., more similar distances) mean an approach is more prone to decoding errors.

Figure 3 reports the results. In all cases, the uncertainty ratio is lower under the ResetBased approach—the decrease ranges from 29.08% (B_4, B_2) up to 96.33% (B_1, B_3). *In practice, the worse uncertainty ratios obtained under the Baseline approach lead to actual decoding errors*—for instance, in some case, when given a sequence that encoded B_1, the Baseline approach wrongly decoded it as B_2, and *vice-versa.* The lower uncertainty ratios under the ResetBased approach imply a generally reduced likelihood of decoding errors, further emphasizing its practical advantages for reliable data transmission in environments where light sensor data is utilized for covert communication. Notably, under the ResetBased approach, we achieved 0% errors even when the smartphone was at maximum distance from the monitor, demonstrating an effective range achievable with a small monitor.

To confirm statistical significance of our experimental results, we performed a *Mann-Whitney U test.*[6] The results of the test showed evidence

[6]The Mann-Whitney U test is a non-parametric test that does not require the assumption of normality. It assesses whether one of two groups of independent observations tends to have larger values than the other. In our case, this test was appropriate to compare the median distances from the correct encoding sequences between the two approaches,

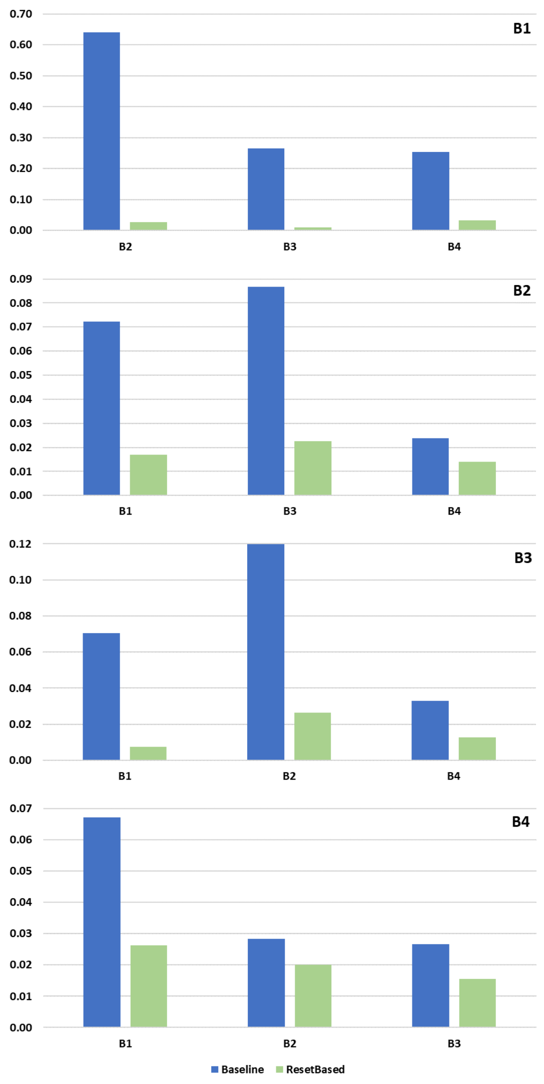


Figure 3: Uncertainty ratios (lower is better).

of statistical significance with a p-value of 0.029—distances were therefore indeed smaller (better) under the ResetBased approach. This statistically

given the data distribution characteristics observed. We did not employ a t-test because it requires data to be approximately normally distributed to provide reliable results, which was not the case for all our datasets according to the Shapiro-Wilk test results.

supports the superiority of the ResetBased approach over the Baseline approach in terms of reducing the average distance for correct byte encodings, corroborating our initial observations from descriptive statistics and the calculated uncertainty ratios.

It should be observed that metrics such as the average distance and uncertainty ratio provide a practical measure of how variability in sensor readings affects the accuracy of our communication scheme. These metrics allow us to quantify the likelihood of errors in a way that reflects the real-world conditions under which the system operates. Furthermore, they account for the average effects of both sensor behavior and environmental factors, offering a robust representation of performance. A purely theoretical analysis of decoding errors appears difficult given the very unpredictable conditions we operate in.

In fact, the core challenge we faced in measuring the decoding error directly stems from the nature of how Android's framework handles sensor data. The light sensor readings are subjected to averaging by the Android system, which introduces uncertainty that is difficult to model theoretically. The readings are not taken at precise or consistent intervals because they depend on when the listener receives sensor data events from the framework. This inherent variability would make it challenging to define a deterministic light sensing function for theoretical analysis. Since the Android framework controls the sensor event system, the timing and frequency of readings are not in our direct control, and hence, predicting the exact decoding error theoretically becomes impractical.

In addition, environmental noise, such as ambient light changes, significantly affects the light sensor readings. The unpredictability of environmental factors means that the luminosity values received by the sensor may fluctuate unexpectedly, even in controlled conditions. This noise would further complicate a theoretical approach to analyzing decoding errors, as any theoretical model would need to account for the variability introduced by these external factors. The noise can drastically alter the readings, especially when dealing with transitions between different luminosity levels, as discussed in Section 3.4.

6. Use Cases

In this section, we discuss possible use cases of our proposed techniques for light-based out-of-band covert communications.

6.1. Cross-Device Tracking

Cross-device tracking employs a variety of techniques to monitor user activity across various platforms, including smartphones, PCs, and smart TVs. It is based on the use of trackers, which can be unique identifiers, cookies, or even signals, allowing advertisers to create a comprehensive user profile by gathering data from multiple devices rather than just one. A notable technique involves the use of audio beacons emitted by one device and captured by the microphone of another. These profiles are crucial for the customization and optimization of targeted advertising strategies, enabling advertisers to deliver content based on their activity across different devices.

Similarly to the deployment of ultrasonic beacons seen in ultrasound cross-device tracking [11], we consider an advertising ecosystem whose logical structure is shown in Figure 4. The process begins when an advertiser,

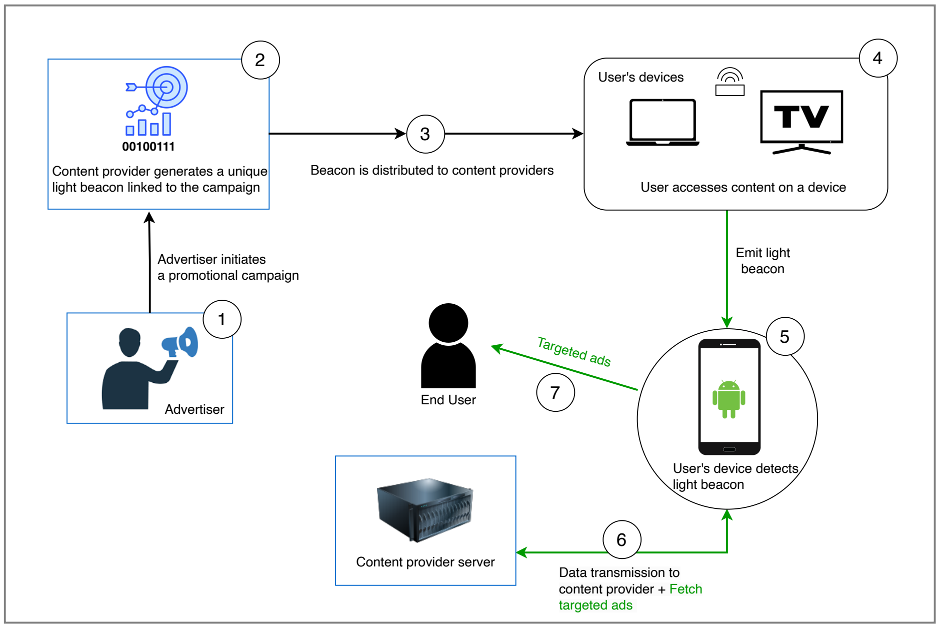

2
00100111
Content provider generates a unique light beacon linked to the campaign
3
Beacon is distributed to content providers
4
User's devices
TV
User accesses content on a device
Advertiser initiates a promotional campaign
1
Advertiser
Emit light beacon
End User
Targeted ads
7
5
User's device detects light beacon
Content provider server
6
Data transmission to content provider + Fetch targeted ads

Figure 4: Interaction between the entities of the mobile advertising ecosystem during a typical cross-device tracking scenario.

possibly a retail brand aiming to market its products or services, initiates a promotional campaign and entrusts the campaign content to the provider (Step 1). The provider generates a distinct light beacon tied to the campaign. This beacon, characterized by light intensity or pattern variations invisible to users, is linked to the campaign (Step 2). Following beacon creation, the provider distributes this beacon alongside the client's advertisements to content providers (Step 3). To maximize the reach, a single beacon might be shared with multiple content providers for efficiency. When a user accesses content from one of these providers on a device, the content, along with the beacon, is delivered to the user (Step 4). As the user interacts with the content, the embedded light beacon is emitted from the screen and captured by the light sensor of another device owned by the user (Step 5).

The main application receives the light beacon and sends the beacon information back to the provider's backend system. This enables the advertising framework to precisely target advertisements based on the user's interests (Step 6). Consequently, the application-equipped device presents advertisements that closely align with the user's preferences (Step 7). This targeted approach increases the likelihood of the user engaging with the advertisement, visiting the advertiser's online or physical store, and ultimately making a purchase.

Security and Privacy Considerations. While the use of light beacons in cross-device tracking enables efficient ad targeting, it also introduces security and privacy risks. The main concerns include the possibility of covert data transmission and unauthorized signal interception, as light-based communication can occur without the user's knowledge. Additionally, nearby devices could potentially intercept light signals, leading to privacy violations or targeted attacks. In the following sections, we provide a more thorough exploration of the security implications of cross-device tracking, by discussing how light-based covert channels can be exploited in various attack scenarios.

6.2. Deanonymization

In this attack scenario, an adversary aims to compromise the anonymity of users by leveraging networks or systems designed for anonymity, such as Tor or VPNs, through the exploitation of light sensor technology. The goal is to identify individuals who wish to remain anonymous while browsing the web. The procedure for this light sensor-based deanonymization is depicted in Figure 5.

First, the adversary creates a trap such that a specific sequence of bytes is encoded as a sequence of luminance values. This "beacon trap" is integrated into web content as discussed in Section 4.4 and in such a manner that, when displayed on a device's screen, it emits the desired light pattern (Step 1).

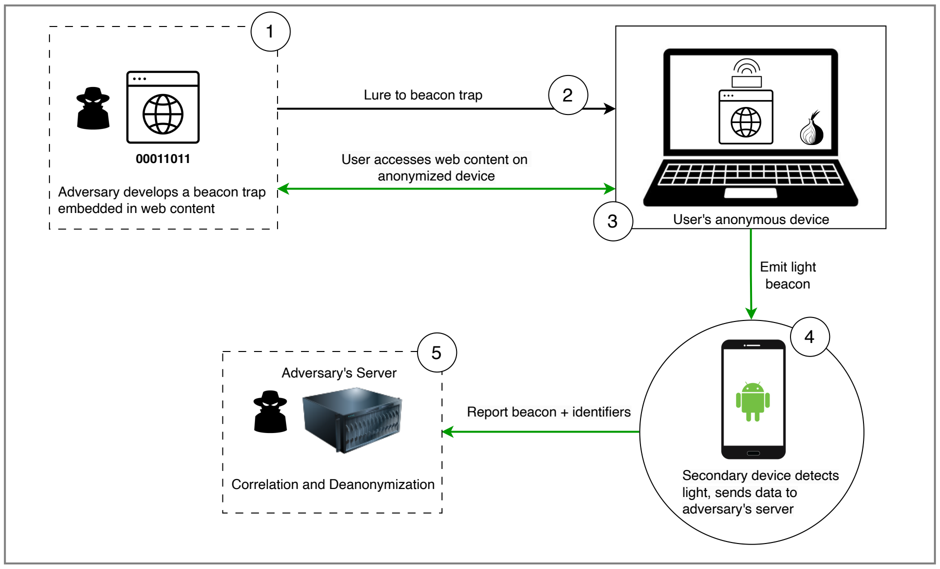

1
00011011
Adversary develops a beacon trap embedded in web content
Lure to beacon trap
2
User accesses web content on anonymized device
3
User's anonymous device
Emit light beacon
4
Secondary device detects light, sends data to adversary's server
Report beacon + identifiers
5
Adversary's Server
Correlation and Deanonymization

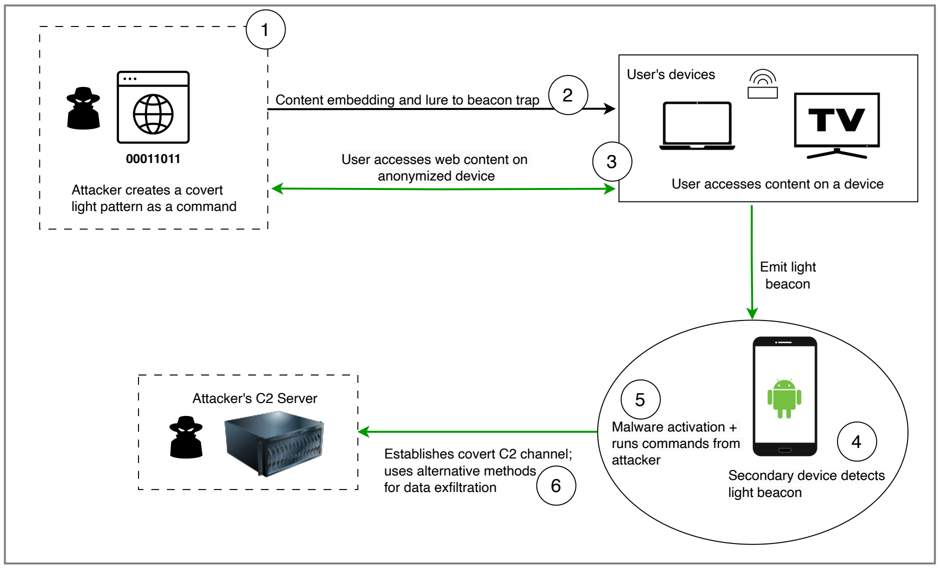
Figure 5: Steps of a light sensor-based deanonymization attack against a user of an anonymized network.

The subsequent action involves the deployment of a strategy to persuade the anonymous user to access the web content containing the trap. This could be achieved using social engineering tactics or by embedding the beacon in some web content frequently visited by the target, such as a particular YouTube video (Step 2). Next, when the target user interacts with the content hosting the beacon trap using their anonymized device, such as a laptop equipped with Tor, the device emits the encoded light pattern (Step 3).

In parallel, a secondary device owned by the target user, which is not protected by the anonymity network yet is equipped with a light sensor and positioned near the primary device, detects the emitted pattern. An application on this secondary device processes the light pattern, converting it into a sequence of readings, and transforming it into a sequence of bits, which is then forwarded to a server under the control of an adversary along with a unique identifier of the user or device (Step 4). In the final step, the adversary correlates the anonymized browsing activity with the non-anonymized secondary device based on the transmitted data. This correlation allows for the deanonymization of the user's activity (Step 5).

6.3. Command and Control

In the Command and Control scenario, adversaries utilize light sensor technology to activate pre-installed malware in targeted devices. An example of a command and control attack scenario is reported in Figure 6.



1
00011011
Attacker creates a covert light pattern as a command
Content embedding and lure to beacon trap
2
User's devices
TV
User accesses web content on anonymized device
3
User accesses content on a device
Emit light beacon
Attacker's C2 Server
5
Malware activation + runs commands from attacker
4
Secondary device detects light beacon
Establishes covert C2 channel; uses alternative methods for data exfiltration
6

Figure 6: Steps of a command and control attack scenario involving light sensor technology.

To perform the attack, the initial step involves the creation of a distinct light pattern by the attacker, that serves as the covert activation command for the embedded malware (Step 1). Again, note that this pattern is strategically designed to go unnoticed by the user. Following this, the attacker embeds the pattern into various forms of digital content, such as videos or interactive advertisements, which are then disseminated across some platform waiting for the interaction of the targeted victim (Step 2). As the victim engages with the infected content on a device (e.g., a personal computer or smart TV), the device's display begins to emit the encoded light pattern (Step 3). This pattern is detected by the light sensor of the target device, which has been previously compromised and lies in wait for this specific activation signal (Step 4). The detection of the light pattern triggers the malware to awake from its dormant state and initiate its intended malicious functions, ranging from unauthorized data exfiltration to propagating the infection to other connected systems (Step 5). The final stage of the attack allows the adversary to establish a covert command and control channel over the infected device. Herein, the light sensor serves as a stealth out-of-band CC for sending activation signals and commands from the attacker to the compromised device. However, an additional CC is necessary for transmitting data back to the attacker. For this purpose, the malware could leverage other mechanisms, such as encoding data into benign network traffic or using acoustic signals undetectable by humans but capturable by the attacker's receiving devices positioned in proximity of the targeted device (Step 6).

6.4. Information Leakage

In this scenario (Figure 7) the attacker exploits the light sensor technology to detect light patterns that reveal information about the user's online behavior and preferences, thereby compromising their privacy.

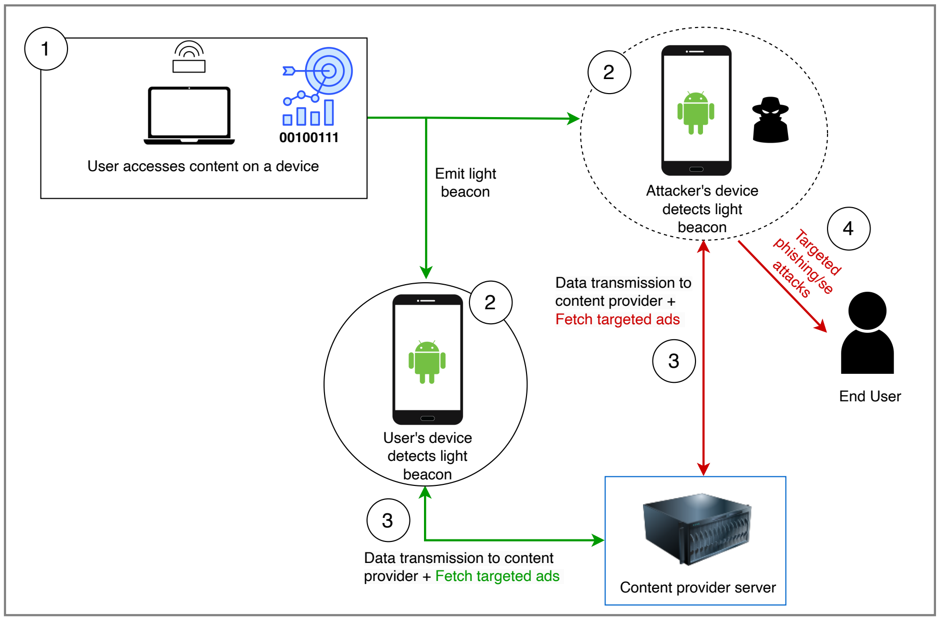

1
00100111
User accesses content on a device
Emit light beacon
2
Attacker's device detects light beacon
4
Targeted phishing/se attacks
End User
Data transmission to content provider + Fetch targeted ads
3
2
User's device detects light beacon
3
Data transmission to content provider + Fetch targeted ads
Content provider server

Figure 7: Steps of the information leakage attack scenario.

To do this, we assume that the attacker is located in physical proximity to the targeted victim. In the initial step of the attack, the targeted user engages in Internet browsing on a computer. During the session, several targeted ads with unnoticeable light patterns are displayed on the victim's screen (Step 1). These ads are tailored to the user's general interests and online activities. Simultaneously, the attacker is handling a device which captures the light patterns and relays them to the service specialized in cross-device tracking (Step 2). Over time, the attacker will collect a sufficient number of patterns and their profile will look very similar to that of the victim. In the final stage, the attacker will start to get served ads that align with the victim's interests (Step 3). Leveraging this inferred information, the adversary can execute targeted attacks, such as phishing or social engineering campaigns (Step 4).

7. Conclusions

In this paper, we have explored the security and privacy implications of light sensor data access on Android devices. After tackling significant challenges that derive from how the Android Sensor Framework works and how sensor data varies, we proposed novel techniques for creating a light-based out-of-band covert channel. Our approaches employ luminance values to encode and decode data, also addressing issues related to environmental factors. Through our experimental assessment, we demonstrated the practicality and effectiveness of using modulated luminance values for data transmission, achieving promising results that validate the feasibility of our methods in real-world scenarios. Hopefully, our study will contribute to the development of effective security countermeasures in the mobile ecosystem, an immediate and useful mitigation action could consist of including the light sensor in the Android permission model, to avoid unrestricted access by any app installed on Android devices.

References

[1] Aviv, A.J., Sapp, B., Blaze, M., Smith, J.M., 2012. Practicality of accelerometer side channels on smartphones, in: Zakon, R.H. (Ed.), 28th Annual Computer Security Applications Conference, ACSAC 2012, Orlando, FL, USA, 3-7 December 2012, ACM. pp. 41–50.

[2] Carrara, B., Adams, C., 2016. Out-of-band covert channels - A survey. ACM Comput. Surv. 49, 23:1–23:36.

[3] Checkoway, S., McCoy, D., Kantor, B., Anderson, D., Shacham, H., Savage, S., Koscher, K., Czeskis, A., Roesner, F., Kohno, T., 2011. Comprehensive experimental analyses of automotive attack surfaces, in:

20th USENIX Security Symposium, San Francisco, CA, USA, August 8-12, 2011, Proceedings, USENIX Association.

[4] Deshotels, L., 2014. Inaudible sound as a covert channel in mobile devices, in: Bratus, S., Lindner, F.F. (Eds.), 8th USENIX Workshop on Offensive Technologies, WOOT '14, San Diego, CA, USA, August 19, 2014, USENIX Association.

[5] Hasan, R., Saxena, N., Haleviz, T., Zawoad, S., Rinehart, D., 2013. Sensing-enabled channels for hard-to-detect command and control of mobile devices, in: Chen, K., Xie, Q., Qiu, W., Li, N., Tzeng, W. (Eds.), 8th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '13, Hangzhou, China - May 08 - 10, 2013, ACM. pp. 469–480.

[6] Kune, D.F., Backes, J.D., Clark, S.S., Kramer, D.B., Reynolds, M.R., Fu, K., Kim, Y., Xu, W., 2013. Ghost talk: Mitigating EMI signal injection attacks against analog sensors, in: 2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013, IEEE Computer Society. pp. 145–159.

[7] Lampson, B.W., 1973. A note on the confinement problem. Commun. ACM 16, 613–615.

[8] Maiti, A., Jadliwala, M., 2019. Light ears: Information leakage via smart lights. Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 3, 1–27.

[9] Marquardt, P., Verma, A., Carter, H., Traynor, P., 2011. (sp)iphone: decoding vibrations from nearby keyboards using mobile phone accelerometers, in: Chen, Y., Danezis, G., Shmatikov, V. (Eds.), Proceedings of the 18th ACM Conference on Computer and Communications Security, CCS 2011, Chicago, Illinois, USA, October 17-21, 2011, ACM. pp. 551–562.

[10] Matyunin, N., Szefer, J., Katzenbeisser, S., 2018. Zero-permission acoustic cross-device tracking, in: 2018 IEEE International Symposium on Hardware Oriented Security and Trust (HOST), IEEE. pp. 25–32.

[11] Mavroudis, V., Hao, S., Fratantonio, Y., Maggi, F., Kruegel, C., Vigna, G., 2017. On the privacy and security of the ultrasound ecosystem. Proc. Priv. Enhancing Technol. 2017, 95–112.

[12] Petracca, G., Sun, Y., Jaeger, T., Atamli, A., 2015. Audroid: Preventing attacks on audio channels in mobile devices, in: Proceedings of the 31st Annual Computer Security Applications Conference, Los Angeles, CA, USA, December 7-11, 2015, ACM. pp. 181–190.

[13] Rouf, I., Miller, R.D., Mustafa, H.A., Taylor, T., Oh, S., Xu, W., Gruteser, M., Trappe, W., Seskar, I., 2010. Security and privacy vulnerabilities of in-car wireless networks: A tire pressure monitoring system case study, in: 19th USENIX Security Symposium, Washington, DC, USA, August 11-13, 2010, Proceedings, USENIX Association. pp. 323–338.

[14] Seyedkazemi, S., Gursoy, M.E., Saygin, Y., 2024. Luxtrack: activity inference attacks via smartphone ambient light sensors and countermeasures. IEEE Internet of Things Journal .

[15] Subramanian, V., Uluagac, A.S., Cam, H., Beyah, R.A., 2013. Examining the characteristics and implications of sensor side channels, in: Proceedings of IEEE International Conference on Communications, ICC 2013, Budapest, Hungary, June 9-13, 2013, IEEE. pp. 2205–2210.

[16] Uluagac, A.S., Subramanian, V., Beyah, R.A., 2014. Sensory channel threats to cyber physical systems: A wake-up call, in: IEEE Conference on Communications and Network Security, CNS 2014, San Francisco, CA, USA, October 29-31, 2014, IEEE. pp. 301–309.