BACON: Improving Broadcast Audio Authentication

Theo Gasteiger*, Pericle Perazzo†, Markus Schuß*, Carlo Alberto Boano*, Fikret Basic*, Gianluca Dini†, Kay Römer*
*Institute of Technical Informatics, Graz University of Technology, Austria
†Department of Information Engineering, University of Pisa, Italy
Email: {gasteiger, markus.schuss, cboano, basic, roemer}@tugraz.at, {pericle.perazzo, gianluca.dini}@iet.unipi.it

***Abstract*—Bluetooth's LE Audio, particularly broadcast audio, is at the forefront of becoming the de facto standard for immersive audio applications in public venues. Nevertheless, the security of the transmitted audio data is solely based on a passkey (`Broadcast_Code`) shared between all (including possibly malicious) receiver devices, leaving many envisaged applications vulnerable to impersonation as well as denial-of-service attacks. In order to address these vulnerabilities, we present BACON, a novel Bluetooth core specification-compliant mechanism for the authentication of Broadcast Isochronous Groups (BIGs). Authenticated BIGs are able to provide data authenticity for broadcast isochronous streams as well as control subevents used to disseminate control information to all receiver devices in the communication range. With BACON, we are the first to outline a mechanism that protects against attacks on broadcast audio applications while being small enough to fit on resource-constrained devices thanks to the underlying protocol's lightweight symmetric cryptography.**
***Index Terms*—Bluetooth Low Energy, Digital audio broadcasting.**

I. Introduction

The Bluetooth core specification (BC) has undergone numerous improvements in recent years, not only enhancing the performance and reliability through the addition of physical modes, but also expanding the range of potential applications with the introduction of novel advertisement schemes, such as periodic advertising [1]. Notably, the BC v5.2 is considered a milestone in Bluetooth history due to the introduction of the so-called *isochronous channels*, laying the foundation for time-bounded data transmission as well as synchronized data rendering across multiple receiver devices [2]. In fact, isochronous channels enable the development of the new LE Audio standard, allowing for use cases such as "true wireless earbuds" [2], [3]. While manufacturers previously had to ensure that audio data transmitted to each earbud individually was rendered at exactly the same time on both earbuds, they can now utilize LE Audio to enforce the strict timing requirements. Moreover, isochronous channels support both connection-oriented (bidirectional) as well as connection-less (unidirectional) communication. The latter, also referred to as a Broadcast Isochronous Stream (BIS), is used for simultaneous data dissemination to an unlimited number of receiver devices within communication range [1].

LE Audio in public settings. Marketed as *Auracast*, LE Audio's broadcast audio feature utilizing BISes enables a plethora of novel use cases that could revolutionize the way people experience the world [2], [4]–[6]. Besides applications in private environments, such as TVs broadcasting audio data to earbuds and hearing aids, BISes foster the creation of numerous unique applications in public spaces, ultimately enabling the creation of immersive listening experiences [5]. In fact, audio diffusion utilizing BISes is foreseen in bars, museums, theaters, conference centers, and transport hubs, enabling people to experience their surroundings in a personalized manner [2], [4], [6]–[9]. For example, airports or train stations may broadcast gate announcements, public transport disruption information, or navigation instructions to all passengers [2], [4], [9]. Museums or exhibitions may replace traditional audio guides with Auracast, enabling visitors to use their own devices to receive the tour information [5], [7]. Even mouth-worn hearing devices based on bone conduction may use this technology to disseminate critical information in tactical or rescue environments [10], [11].

Authenticity of BISes in public venues. Although Auracast recommends the usage of open (unencrypted) BISes in public spaces [12], many of the aforementioned use cases rely on the authenticity of the audio stream. Indeed, attackers may take advantage of unprotected audio streams and cause injury (e.g., by manipulating navigation information for visually-impaired people), or undertake defacement actions (e.g., by replacing the original audio in theaters or airports with political or defamatory messages). To defend against these threats, the Bluetooth Special Interest Group (SIG) foresees stream authentication in the form of private (encrypted) BISes. Utilizing the so-called `Broadcast_Code` (i.e., a passkey used to derive a symmetric session key), private BISes are able to not only provide audio confidentiality, but also prevent audio manipulation. However, such protection mechanisms rely on the secrecy of the `Broadcast_Code`, which is hardly guaranteed in a public setting. In general, the distribution of the `Broadcast_Code` should be "easy to do" [2] and is envisaged to be executed utilizing out-of-band methods such as NFC tags or QR codes mounted at prominent locations [1], [2]. For example, passengers in an airport can simply tap NFC-enabled headphones to an LE Audio NFC tag located at the gate or scan a QR code displayed on a TV in a dedicated area [8]. Unfortunately, if authorized passengers are able to do so, so are malicious actors, who can, therefore, easily gain access to the used key material.

As demonstrated by the BISON attack [13], malicious actors can exploit this weakness and not only impersonate a broadcasting device, but also take over an ongoing, encrypted BIS. From a security perspective, this predicament is alarming, as it demonstrates that BISes can easily be manipulated, potentially resulting in injury or harm. While digital signatures may address the problem, many envisaged applications using resource-constrained devices (e.g., earbuds or hearing aids) are incapable of performing the necessary signature verification operations within the stringent timing constraints imposed by BISes [14].

Contributions. In this paper, we present BACON, a novel and BC-compliant mechanism enabling the authentication of BIG subevents in the presence of adversaries knowing the `Broadcast_Code`. BACON achieves this by adapting the Timed Efficient Stream Loss-tolerant Authentication (TESLA) protocol [15] to BISes, in order to guarantee authentication by utilizing only hash functions and symmetric-key encryption. The co-presence of TESLA and BIS robustness mechanisms such as payload pre-transmission raises peculiar and novel problems, which we address in this paper. By employing BACON, we can secure BISes against attacks such as BISON [13], while allowing ordinary BLE devices (i.e., devices not employing BACON) to seamlessly receive the original audio stream.

II. Broadcast Isochronous Streams (BISes)

BISes can be described as an ordered series of subevents sent at fixed intervals, forming a stream of data with strict timing requirements. As illustrated in Fig. 1, packets belonging to the same BIS (e.g., BIS_{A1} and BIS_{A2}) constitute a BIS event (BIS Event_A). One or more BIS events as well as an optional control subevent (CTRL) make up a Broadcast Isochronous Group (BIG). In turn, BIG events are sent at regular intervals (ISO interval), creating the actual isochronous data stream. The optional control subevent is used to disseminate control information, such as the indication to terminate the ongoing stream or to switch to a different channel map.

Robustness of BISes. In order to increase resilience against RF interference, BISes employ multiple strategies to maximize the probability of receiving the transmitted data [1], [2]. One strategy involves the retransmission of subevents to accommodate the lack of acknowledgments. This is controlled with the so-called retransmission number (RTN). For example, RTN=2 denotes two additional retransmissions of the given subevent. The overall number of subevents in each BIS event is called the number of subevents (NSE). Depending on the number of unique payloads in each BIS event, BIS subevents may be transmitted in bursts. For example, one BIS event carrying a large payload may require fragmenting the payload into three smaller payloads (i.e., one unique payload consisting of three subevents is considered a burst). Hence, the BC introduces the burst number (BN), i.e., the number of new payloads provided within each BIS event [2]. In turn, the ratio between the NSE and the BN is considered the group count (GC), which determines the arrangement of retransmissions in a BIS [2]. Two examples of a BIS utilizing retransmissions of a large payload (i.e., NSE=6, BN=3, fragmented into three payloads and NSE=6, BN=2, fragmented into two payloads) are illustrated in Fig. 2.

Another strategy to increase the robustness of BISes involves the usage of pre-transmissions, i.e., subevents associated with multiple points in time in order to increase transmission scheme diversity [2]. To this end, the BC foresees the usage of the immediate repetition count (IRC) and the pre-transmission offset (PTO) [1]. As exemplified in Fig. 3, the IRC defines the number of data groups belonging to the current BIS event, whereas the PTO determines the size of the increment (in terms of BIG events) from which the data group originates. Denoting a data group as g_n, the offset to this origin O_{g_n} can be computed as:

$$O_{g_n} = (n - IRC + 1) \cdot PTO \quad \forall n \in [IRC, NSE - 1] \quad (1)$$

For example, as illustrated in Fig. 3 (bottom), IRC=2 defines that two out of four data groups (with four being the GC) belong to the current BIS event, whereas the remaining two data groups (GC - IRC) belong to future BIS events.

Depending on the PTO, the corresponding future BIS events occur O_{g_n} BIG events after the current BIG event. In Fig. 3 (bottom), where PTO=2, the data group g_2 in the first BIG event (BIG0) has its origin in BIG2 ($O_{g_n} = (2 - 2 + 1) \cdot 2 = 2$), whereas g_3 has its origin in BIG4 ($O_{g_n} = (3 - 2 + 1) \cdot 2 = 4$). Therefore, the combination of IRC and PTO allows the early transmission of data groups belonging to future BIS events, ultimately enhancing the robustness of BISes [2].

Rendering of isochronous data. Isochronous communication allows for the simultaneous presentation of audio data across multiple devices. Devices using this technology also incorporate the so-called presentation delay [2]. This user-selectable delay introduces an artificial point in time, known as the rendering point (RP), at which all receiver devices must be ready to present the received data including any additional processing such as active noise cancellation or hearing aid audio adjustments in case of audio-focused applications [2].

BIS security. BISes can be unencrypted and unauthenticated, encrypted and unauthenticated, or encrypted and authenticated. All these cryptographic operations are based on the `Broadcast_Code`, which is a passkey used to derive a symmetric session key. In turn, this session key is used to not only encrypt the payload of all BIG subevents, but also to authenticate them with a 32-bit message authentication code, referred to as message integrity check (MIC) in the BC [1].

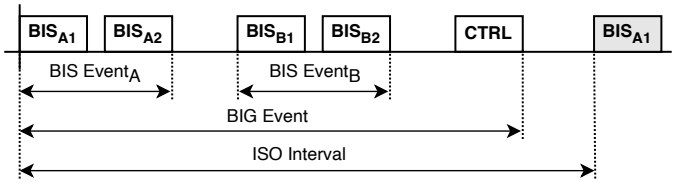


Fig. 1: **BIG encapsulating two BISes in sequential order.** Every BIG event contains one or more BIS events (e.g., BIS Event_A), in turn consisting of one or more subevents (e.g., BIS_{A1}, BIS_{A2}) as well as an optional control subevent (CTRL).

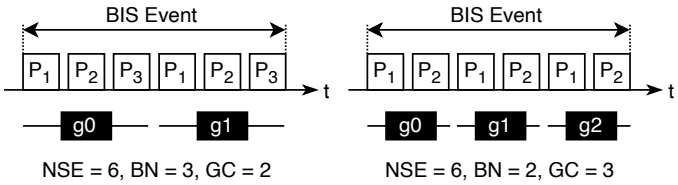
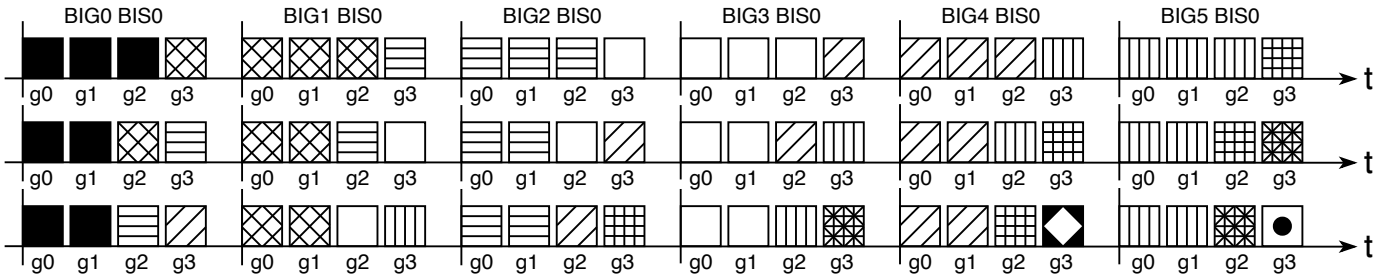


Fig. 2: **Impact of the group count on a BIS event consisting of six subevents.** Left: Three unique payloads (p1, p2, p3) are sent twice. Right: Two unique payloads (p1, p2) are sent thrice.



TOP: NSE = 4, BN = 1, IRC = 3, PTO = 1, GC = 4 **MIDDLE:** NSE = 4, BN = 1, IRC = 2, PTO = 1, GC = 4 **BOTTOM:** NSE = 4, BN = 1, IRC = 2, PTO = 2, GC = 4

Fig. 3: **Impact of the IRC and PTO on BIS events.** Data associated with each BIS event is transmitted in the first (IRC) slots. Data associated with future BIS events is transmitted in the remaining (GC - IRC) slots, offset utilizing the PTO.

III. Threat Model

While the BC foresees encrypted and authenticated BISes, such protection mechanisms are ineffective against adversaries knowing the `Broadcast_Code`, leaving many LE Audio use cases vulnerable to impersonation or denial-of-service attacks.

Overshadowing of BIG subevents. Malicious actors can exploit the capture effect [16] to transmit forged packets while being in close proximity to the receiver device or when having the ability to send at a high transmission power (i.e., overshadow the legitimate signal). As a result, attackers may be able to impersonate the legitimate stream or even be able to execute a denial-of-service attack by forging BIS packets and transmitting a stream termination control subevent.

Hijacking ongoing BISes utilizing BISON. A particularly concerning attack exploiting BIG control subevents is the so-called BISON attack [13]. This attack leverages the vague specification of the `Broadcast_Code` exchange (i.e., the fact that the `Broadcast_Code` is envisaged to be distributed in the form of QR codes or NFC tags deployed in public settings) in combination with BIG channel map updates (intended to improve the link quality by blacklisting specified frequency channels in noisy environments) to hijack an ongoing BIS [1], [2]. As a result, receiver devices are fooled into thinking they are receiving a legitimate BIS, while the attacker controls the actual BIS, causing disruption among synchronized devices.

Threat mitigation utilizing broadcast authentication. In order to reduce the attack surface imposed by BISes, customary authentication can be used. However, as highlighted in §II, the authentication mechanism introduced in the BC may be exploited by malicious actors who gain access to the `Broadcast_Code`, rendering this authentication mechanism ineffective. An alternative to this authentication mechanism is the use of digital signatures. While this provides authenticity, it is also important to consider the application environment. Most receiver devices (e.g., earbuds or hearing aids) have limited resources and are incapable of performing computationally complex operations within the strict timing requirements imposed by isochronous communication [17]. As a result, we propose the usage the TESLA protocol [15], as a building block to enable the authentication of BIG subevents while minimizing the introduced communication and computation overhead.

IV. TESLA Broadcast Authentication

The Timed Efficient Stream Loss-tolerant Authentication (TESLA) protocol [15], [18] provides broadcast authentication based on an iteratively applied one-way hash function in combination with message authentication codes (MACs).

Sender setup. In order to enable the application of the TESLA protocol, sender devices must first split time into uniform intervals of duration T_{int}. Each time interval is then assigned a self-authenticating value, also referred to as "key", generated by iteratively applying a one-way hash function $H(\cdot)$ to an initial random seed k_n. In detail, the keys are computed as $k_i = H(k_{i+1})$ for each $i \in [0, n-1]$, and form a hash chain of length n. The sender produces the keys in a decreasing-index order (i.e., $k_n, k_{n-1}, \ldots, k_0$), and subsequently uses them in an opposite manner ($k_1, k_2, \ldots, k_n$). Given that the hash chain must also be stored at the sender side, some efficient methods [19] to do so have already been proposed, requiring only $\mathcal{O}(\log n)$ space, and $\mathcal{O}(\log n)$ time to reconstruct each key.
After the setup, the sender can start the transmission of messages, each of which must be accompanied by a MAC computed with the key associated to the current time interval. During the i-th time interval, the sender also reveals the key that it used to compute MACs in the previous $(i-d)$-th time interval, with $d \geq 1$ being the "key disclosure delay".

Message authentication. Each receiver must know the time instants at which the keys will be disclosed ("key disclosure schedule"), and a *commitment* to the hash chain, typically k_0. It is crucial that the commitment is exchanged over an authenticated channel in order to prevent impersonation attacks. Moreover, each receiver must have a (possibly internal) trusted time reference, which must be loosely synchronized[1] with the sender, with a bounded clock skew [15]. The receiver does not authenticate the messages immediately, but rather buffers them to enable their subsequent authentication. When the receiver receives a new message and its MAC, it first determines whether the corresponding key has already been disclosed by the sender. To do this, the receiver uses its knowledge on the key disclosure schedule and its trusted time reference. If the key has been (or may have been) disclosed, the receiver discards

[1]Sender and receiver devices must have the same (loose) notion of time in order to mitigate replay attacks (e.g., re-use of yesterday's hash chain).

the message as untrusted. Otherwise, it buffers the message and its MAC, waiting for the corresponding key to be disclosed. When disclosed, the receiver checks that it actually makes part of the hash chain by computing its hash and comparing it with the previous key. If the check is successful, the receiver stores the key in order to be able to authenticate successive keys. Moreover, if a message MACed with that key was buffered, the receiver verifies its MAC. If the MAC is correct, the message is considered authentic, which means that the receiver unbuffers the message and passes it to the higher-layer protocol. TESLA is tolerant to message losses, since if the receiver misses one or more key disclosures, it can "reconnect" to the hash chain afterwards, as soon as it receives a new key disclosure.

TESLA parameters. T_{int}, d and n must be tailored to obtain specific performance tradeoffs. In particular, too large T_{int} and d lead to extended buffering on receiver devices. Too small T_{int} and d cause the receiver to discard many legitimate messages, especially if the transmission delay is high or the clock skew between the sender and the receiver is pronounced. A too large n forces the sender to create long hash chains, thus slowing down the setup phase. Conversely, a too small n causes the hash chain to be rapidly depleted, forcing frequent sender setups.

Security. It is important to consider the key and MAC sizes to ensure a sufficient level of security while minimizing the incurred overhead. If keys are too short, malicious actors may attempt to brute-force a future key, by breaking the preimage resistance of the hash function (key-guessing attack). If successful, attackers are able to forge authenticated messages until the guessed key is revealed by the legitimate sender. On the other hand, if MACs are too short, attackers may randomly guess the correct MAC of a forged message, and send the message and the MAC to the receiver (MAC-guessing attack).

V. BACON: Broadcast Audio Authentication

As the disclosed vulnerabilities in §III highlight, authentication of BISes as well as control subevents in the presence of an adversary knowing the `Broadcast_Code` should be considered indispensable for future BC revisions. Therefore, we present the BACON mechanism, which enables the authentication of BIG subevents, while incurring little computational overhead, and maintaining compatibility with BC v5.2 or higher. Please note that, while fully compliant to the BC, BACON requires access to the BLE controller via vendor-specific HCI extensions[2], as knowledge about the robustness parameters outlined in §II is needed to avoid premature disclosure of the keys (see Phase 3).

Design. TESLA can be adapted for BISes in many ways. One option is to include the MAC and key in the BIS carrying the audio data. However, this creates an issue for LE Audio receivers, which only expect encoded audio data. Another option is to use the control subevent (CTRL) by extending its format, ultimately breaking BC compliance. Alternatively, one may repurpose the MIC field in each BIS payload by including the MAC and key required by TESLA. However, this would also violate BC compliance. Therefore, to provide authenticity for BIS payloads and CTRLs, we employ an additional BIS (BACON) to disseminate authentication information, namely the MACs of the BIS payloads, the MAC of the CTRL, and the keys of the hash chain. By accommodating all TESLA-related information in a dedicated BIS, we maintain compatibility with non-BACON receivers, which will interpret the BACON BIS as an additional, undecodable audio source. From now on, we will refer to BACON's MICs as "BICs" to avoid confusion with the BC MICs. Fig. 4 illustrates an exemplary BIG containing three BISes representing three different languages of an immersive audio guide. The fourth BIS (BACON) is dedicated to BIG authenticity and, therefore, contains the BICs of all language BISes and a key of the hash chain. Moreover, the BIG may contain a CTRL to indicate, for example, the use of a different channel map starting at a given time instant. As such the BACON BIS must also include the BIC of the CTRL in addition to the audio BISes should such CTRL be present.

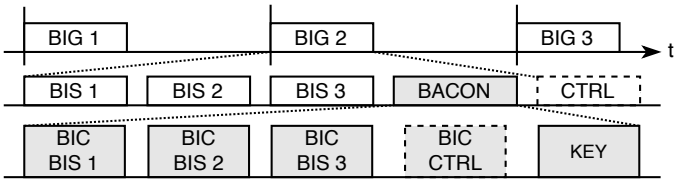


Fig. 4: **BACON-secured BIS.** Every BIG event consists of one or more data BISes (e.g., BIS 1, BIS 2, and BIS 3), a BACON BIS, and an optional control subevent (CTRL). In turn, the BACON BIS contains the BICs of all data BISes, the BIC of the control subevent, and a key of the utilized hash chain.

Phase 1: Broadcaster setup. Like TESLA, BACON uses a pre-computed chain of self-authenticating keys generated by the iterative application of a one-way hash function. Given the use cases presented in §I and the often asymmetric link budget of real-world applications (i.e., transmitter nodes are usually mains-powered and not subject to strict power requirements), this computationally complex phase may be offloaded to more powerful devices fulfilling the isochronous broadcaster role [2]. However, it is still crucial to carefully choose the length of the hash chain depending on the lifetime of the BIS, as it significantly affects not only the memory footprint, but also the on-air time and, thus, the time needed for receiver devices to be powered (and hence their power consumption). Similar to the BC, we suggest the usage of 128-bit keys. However, BACON is also able to handle different key sizes, allowing to trade security for efficiency. For example, a typical LE Audio broadcast application in a public setting may employ a 10 ms ISO interval over a predetermined lifetime of 24 hours. We assume for simplicity that T_{int} is equal to an ISO interval ($T_{int} = 10ms$), in such a way that every BIG event carries a key. Therefore, the broadcaster must precompute a one-way chain of at least $n = 8,640,000$ keys ($\approx$1.52 GB in total). Supposing 128 bits for each key, storing the hash chain with the logarithmic Coppersmith-Jakobsson

[2]The issue is that the RTN provided to the controller via the regular HCI_LE_Create_BIG is only a recommendation to the controller, which internally then derives other relevant robustness parameters.

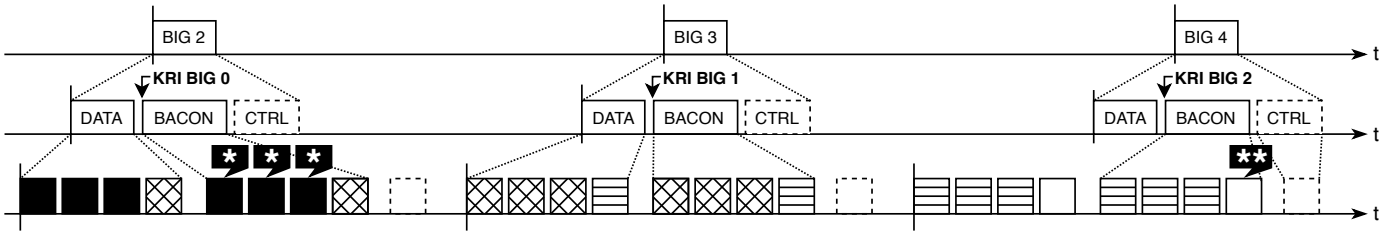


Fig. 5: **Impact of pre-transmissions on the key reveal instant (KRI).** In BACON-secured applications using pre-transmissions with non-fragmented payloads (BN=1), NSE=4 (thus, GC=4), IRC=3, and PTO=1, early key disclosure occurs. BACON mitigates this by shifting the KRI to the $(i+d)$-th BIG event (e.g., BIG2 data is authenticated by the ($\star$) BICs computed with the ($\star\star$) key).

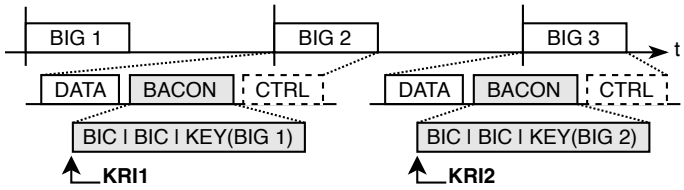


Fig. 6: **KRIs of BACON-secured BISes without pre-transmissions.** BIS events of the i-th BIG event can be authenticated at the akin KRI, located at the $(i+1)$-th BIG event.

method [19] requires the broadcaster to store approximately $(\lceil \log_2 n \rceil + \lceil \log_2(\log_2 n + 1) \rceil) \cdot (128 \text{ bits}) = 464$ bytes, and to compute $\lfloor \log_2 \sqrt{n} \rfloor = 11$ hashes every ISO interval to reconstruct the corresponding key. Given that even a low-cost single-board computer like a Raspberry Pi 5 can perform several millions of SHA256 hash computations per second [20], this overhead is considered negligible. Moreover, the BIC size in the authentication procedure should account for the number of employed BISes. The higher the number of BISes in a BIG, the more significant the impact of the chosen BIC size. We recommend a value between four (in line with the BC MICs) and eight bytes (lower probability of a successful MAC-guessing attack under a modest transmission overhead increase).

Phase 2: Receiver bootstrap. To ensure the secure initial exchange of information, such as the commitment to the one-way chain (i.e., a previously disclosed key), BACON leverages the principles established within the broadcast assistant role [21], a role specifically designed for LE Audio. Broadcast assistants are intermediaries (e.g., smartphones) that not only provide a way for paired receiver devices (e.g., earbuds) to find and join an ongoing audio stream but also allow for the secure initial exchange of information between the isochronous broadcaster and the isochronous receiver. For example, one could build certificate-based authentication on top of Bluetooth's Generic Attribute Profile (GATT) [1] to verify the identity of the isochronous broadcaster and its time reference. BACON utilizes this to initially exchange the commitment to the one-way chain and a possibly trusted time reference, similar to §IV. Once a receiver obtains this information, its bootstrap is complete and it can disconnect from the broadcast assistant.

Phase 3: Message authentication. The broadcasting device computes a BIC for each subevent (except BACON subevents) inside the BIG event, as well as the optional control subevent using the current key. In turn, these BICs are used to generate the BACON BIS which, together with the data BISes, is disseminated to all receiver devices in communication range. In order to avoid trusting an already-revealed key, all devices need to agree on the time instance at which such key is no longer used to sign new data. For this, BACON leverages the concept of key reveal instant (KRI). We refer to the latter as the beginning of the BACON BIS containing the key. In absence of BIS robustness mechanisms, the key k_n used in BIG n is revealed in the BACON BIS sent within BIG $n+1$, as shown in Fig. 6. Therefore, the KRI of such key is the beginning of the BACON BIS sent within the next BIG. In general, after computing the KRI, a receiver must discard a BIS packet whose BIC is received after the relative key's KRI. However, when employing the BIS robustness mechanisms involving pre-transmissions, the early transmission of BACON packets[3] may cause the disclosure of keys that will be used afterwards to compute and send BICs, as illustrated in Fig. 5. This would inevitably cause receivers to discard such BICs and the relative BIS packets as untrusted. Therefore, the KRI needs to be delayed to future BACON BIS events to account for this. We call this key disclosure delay d, where:

$$d \geq (GC - IRC) \cdot PTO + 1. \tag{2}$$

With Eq. 2, we ensure that no key is disclosed before being used to compute a BIC. As highlighted in Fig. 5, by setting the disclosure delay to $d = 2$ (i.e., by shifting the KRIs by two BIG events), we ensure that BACON is able to cope with the BIS robustness schemes. It is important to highlight that, due to Eq. 2, the presence of pre-transmissions compels the usage of a greater key disclosure delay, which in turn shifts the data usability time (i.e., rendering point), assuming the same presentation delay. This suggests that trade-offs can be found between robustness and audio latency.

BICs of control subevents. Unlike regular BIS subevents, CTRLs are not subject to the BIS robustness mechanisms. In case the BACON BIS includes the BIC of the CTRL, its BIC (and key) are transmitted before the actual CTRL. This enables the immediate validation and processing of the CTRL at the moment of reception. Given that the BACON BIS is subject to these robustness mechanisms, the transmission of the CTRL BIC (and key) must be initiated at an offset of d before the transmission of the CTRL. This could, in practice, be done by delaying the first d CTRL transmissions at the controller level.

[3]Disabling pre-transmission of BACON BISes would break BC compliance.

VI. BACON in Action

We demonstrate the effectiveness of BACON on two Nordic Semiconductor nRF52840 development kits, representing an isochronous broadcaster transmitting dummy audio data (i.e., 60 byte LC3-24_2[4] at 0 dBm), and an isochronous receiver, receiving and authenticating the transmitted data packets with BACON. The proof-of-concept implementation, which we release as open-source[5], is based on the Zephyr real-time operating system [22] and modified versions of the *isochronous broadcaster* and *synchronized receiver* samples. We utilize 128-bit keys computed by iteratively applying the truncated SHA-256 digest, and the truncated HMAC-SHA256 digest for the BICs. We implemented such functions with TinyCrypt [23], which, due to already existent integration in the Zephyr RTOS, eliminates the need for flash memory utilization evaluation.

Experimental setup. The devices are spaced one meter apart with a Nordic Semiconductor nRF7002 development kit acting as a Wi-Fi access point in between, to simulate channel interference. The Wi-Fi access point continuously broadcasts 1400 byte packets every 500 µs at +16 dBm, with a 12 Mbit/s data rate on channel six (IEEE 802.11ax). Power measurements are conducted on the receiver device utilizing a Nordic Semiconductor Power Profiler Kit II in ampere meter mode.

Computational overhead. Security operations in BACON, i.e., key and BIC validation, inevitably extend the runtime of standard broadcast audio receiver applications. Therefore, we conduct an experiment utilizing the aforementioned test setup, extended with a logic analyzer to measure (utilizing the GPIO pins) the introduced computational overhead of the used security operations over ten consecutive test runs consisting of 1000 BIG events. Our experiment shows that key validation of a 128-bit key on the receiver side introduces 108 µs (± 18 µs) of computational overhead. Additionally, each BIC validation on the receiver side takes 606 µs (± 15 µs) or 614 µs (± 19 µs) when using four or eight-byte BICs respectively. This highlights that the computational performance of receiver devices is minimally affected by the chosen level of security. In total, if we consider eight-byte BICs, a BN of one, and an always-present CTRL, an earbud receiving a single BIS will incur (108 µs) + 2×(614 µs) = 1.336 ms of computation time per ISO interval.

Robustness. The introduced BACON BIS increases the susceptibility to RF interference, because in order to correctly receive a BIS packet the receiver must also receive the BACON packet carrying its BIC, and the one carrying its key. Consequently, we conduct an experiment using the aforementioned test setup. Ten test runs consisting of 1000 BIG events with a varying number of retransmissions[6] highlight the impact of RF interference on BACON. As illustrated in Fig. 7, zero retransmissions employing an eight-byte BIC result on average in an 11 % decrease in the packet reception rate (PRR) compared to a BIS without BACON. As soon as we start to increase the RTN, the difference in PRR decreases to under 1 %.

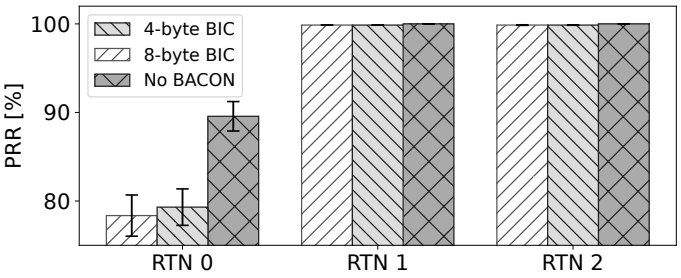


Fig. 7: **Impact of Wi-Fi based RF interference on the packet reception rate.** The reduction in packet reception rate (PRR) due to packet loss, caused by the utilization of BACON, can be counteracted through the application of an increased RTN.

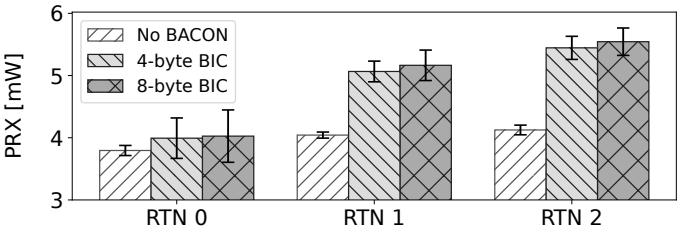


Fig. 8: **Impact of Wi-Fi based RF interference on the power consumption.** Packet loss on BACON-secured BISes forces receiver devices to recompute lost keys. Higher RTN values also increase the probability to stay powered in noisy environments.

Power consumption. RF interference on BACON-secured BISes not only impacts the PRR: due to the increased number of transmitted packets (caused by the BACON BIS), lost BACON packets may force receivers to recompute the lost key to reconnect to the hash chain. For example, in the scenario shown in Fig. 6, missing the BACON BIS subevent in BIG 2 implies that the receiver must wait until the reception of the BACON BIS subevent in BIG 3 and apply the hash function twice to validate the key. Therefore, we conduct an experiment utilizing the aforementioned test setup, consisting of ten test runs with 1000 BIG events. As illustrated in Fig. 8, our experiment shows that Wi-Fi based RF interference on a BACON-secured BIS utilizing four and eight-byte BICs, with an RTN of zero increases the receiver's power consumption (PRX) by ≈ 7 % compared to a test setup employing no authentication. Similarly, an RTN of one increases the power consumption by ≈ 27 %, and an RTN of two by ≈ 33 %. Although receivers switch off after receiving one valid packet (i.e., do not receive all subsequent retransmissions), the higher the RTN, the higher the possibility for receiver devices to stay powered in noisy environments. Hence, a high RTN may also lead to higher power consumption which could affect the lifetime of battery-powered devices.

VII. Discussion and Future Work

Limitations of the implementation. Due to limitations imposed by the Zephyr real-time operating system [22], the current implementation is limited to two BISes, i.e., one BIS carrying dummy audio data and one BIS containing the BACON

[4]Mandated for broadcast audio devices; adequate for music under imperfect listening conditions (e.g., background noise, hearing impairment) [2].

[5]https://iti.tugraz.at/bacon

[6]The RTN setting equals NSE + 1 in the utilized Zephyr version.

information. Future work will include the extension of BACON to multiple BISes and the utilization of real-world audio data.

Scalability of BACON-secured BISes. BACON uses one of the 31 available BISes within a BIG, sacrificing a small amount of on-air time to ensure the authenticity of audio applications. Each additional BIS (e.g., carrying audio) entails the addition of another BIC inside the BACON event. This inevitably affects the size of the transmitted BACON BIS, computational overhead, and therefore leads to an increased power consumption.

Computational overhead. Experiments in §VI show that BACON adds ∼1.335 ms to the runtime of a single-stream broadcast audio receiver application (LC3-24_1), potentially occupying up to 18 % of the available processing time. The remaining processing time must suffice for the intended (audio) use case. It remains to be shown whether this has an impact on practical audio applications, which we will pursue in future work.

Proof-of-concept with audio data. The impact of packet loss on BACON in relation to packet loss concealment algorithms utilized in the mandatory LC3 codec shall be analyzed.

VIII. Related Work

Attacks and countermeasures on Bluetooth Low Energy. In recent years, several BLE vulnerabilities have been disclosed, affecting billions of devices. Attacks such as BtleJack [24] and Ubertooth [25] take advantage of the fact that important metadata such as the access addresses or the preamble are transmitted in plaintext, allowing for the interception of critical information. While BtleJack demonstrates the potential consequences of packet jamming, BISON [13] goes one step further and leverages the obtained information to completely take over an ongoing data stream. Even though SPADE [26] proposes a countermeasure to secure periodic advertisements, which form the basis for isochronous communication, no mitigation to attacks on BIGs and, therefore, BISes has been published yet. We are, to the best of our knowledge, the first to enhance the authentication mechanism for broadcast audio to mitigate impersonation and denial-of-service attacks against BISes in case of an adversary that knows the `Broadcast_Code`.

Applications of the TESLA protocol. Perrig et al. [27] first adapted TESLA to constrained devices for message authentication in wireless sensor networks. The RFC 4383 by Baugher and Carrara [18] adapted the TESLA protocol for SRTP, a network protocol for delivering audio and video over IP networks. The use cases of SRTP are close to the ones of BISes, although SRTP mainly fits remote streaming towards resourceful devices. More recently, the European Union applied TESLA in the Galileo's Open Service Navigation Message Authentication (OS-NMA) protocol [28], to guarantee authentication of GNSS navigation data sent by satellites. We are the first, to the best of our knowledge, to adapt TESLA for authentication of LE Audio broadcasts. It is worth noting that, in BISes, the co-presence of TESLA and robustness mechanisms such as payload retransmission and pre-transmission raises peculiar and novel problems, that we address in this paper, and that are neither addressed by RFC 4383 nor by OS-NMA.

IX. Conclusion

In this paper, we present BACON, a novel and BC-compliant mechanism for authentication of BIG subevents. With BACON, we are the first to apply the principles established within the TESLA protocol to Bluetooth's BISes. We show that BACON can provide data authenticity even with Bluetooth's robustness schemes are applied. We demonstrate the viability of BACON in the context of broadcast audio, utilizing off-the-shelf devices, highlighting the feasibility of BACON under interference.

References

[1] Bluetooth SIG. Bluetooth Core Specification, v5.4. Jan 2023.
[2] Nick Hunn. *Introducing Bluetooth LE Audio*. 2022.
[3] Bluetooth SIG. Bluetooth LE Audio FAQs. https://www.bluetooth.com/media/le-audio/le-audio-faqs. Last access: May 2024.
[4] The OnQ Team. LE Audio: A new age of Bluetooth audio sharing. https://bit.ly/49UeCZa. Last access: May 2024.
[5] Andrew Zignani. LE Audio, Auracast Broadcast Audio, and the Future of Bluetooth Audio. https://bit.ly/42ZbM2N. Last access: May 2024.
[6] Georger Kimathi. Auracast Broadcast Audio: The New Way To Share Audio. https://bit.ly/4bXF6ea. Last access: May 2024.
[7] Bluetooth SIG. Key Use Cases in Public Locations. https://www.bluetooth.com/auracast/public-locations. Last access: May 2024.
[8] Simon Cohen. Auracast looks to radically change how you use Bluetooth. https://bit.ly/4bWKlL5. Last access: May 2024.
[9] Franz Dugand. Unmute the World With Auracast Broadcast Audio. https://bit.ly/3T2hSej. Last access: May 2024.
[10] Audiology Worldnews. Bluetooth communication system commissioned by American military. https://bit.ly/4dlvieH. Last access: May 2024.
[11] Sonitus Technologies. Sonitus Tactical. http://www.sonitustechnologies.com/tactical/. Last access: May 2024.
[12] Jason Marcel. Answers to Commonly Asked Questions About Auracast Broadcast Audio. https://bit.ly/4bWKA8X. Last access: May 2024.
[13] Theo Gasteiger et al. BISON: Attacking Bluetooth's Broadcast Isochronous Streams. In *Proc. of the 20th EWSN Conf.*, 2023.
[14] Peter Kietzmann et al. A Performance Study of Crypto-Hardware in the Low-end IoT. In *Proc. of the 18th EWSN Conf.*, 2021.
[15] Adrian Perrig et al. The TESLA Broadcast Authentication Protocol. *RSA CryptoBytes*, 2002.
[16] K. Leentvaar and J. Flint. The Capture Effect in FM Receivers. *IEEE Transactions on Communications*, 24(5), 1976.
[17] Lena Boeckmann et al. Usable Security for an IoT OS: Integrating the Zoo of Embedded Crypto Components Below a Common API. In *Proc. of the 19th EWSN Conf.*, 2022.
[18] Elisabetta Carrara and Mark Baugher. The Use of Timed Efficient Stream Loss-Tolerant Authentication (TESLA) in the Secure Real-time Transport Protocol (SRTP). RFC 4383, February 2006.
[19] Don Coppersmith and Markus Jakobsson. Almost Optimal Hash Sequence Traversal. In *Financial Cryptography*, 2003.
[20] Michael Larabel. Raspberry Pi 5 Benchmarks: Significantly Better Performance, Improved I/O. https://www.phoronix.com/review/raspberry-pi-5-benchmarks/6, September 2023.
[21] Bluetooth SIG. Basic Audio Profile, v1.0.1. Jun 2022.
[22] Zephyr Project. About the Zephyr Project. https://zephyrproject.org/learn-about. Last access: May 2024.
[23] Zephyr Project. TinyCrypt Cryptographic Library. https://docs.zephyrproject.org/latest/services/crypto/tinycrypt.html. Last access: May 2024.
[24] Damien Cauquil. You'd better secure your BLE devices or we'll kick your butts! https://bit.ly/3P4MUAS. Last access: May 2024.
[25] Mike Ryan. Bluetooth: With Low Energy Comes Low Security. In *Proc. of the 7th USENIX WOOT*, 2013.
[26] Dokyun Ryoo et al. SPADE: Secure Periodic Advertising using Coded Time-Channel Rendezvous for BLE Audio. 2023.
[27] Adrian Perrig et al. SPINS: Security Protocols for Sensor Networks. In *Proc. of the 7th ACM MobiCom*, 2001.
[28] Ignacio Fernández-Hernández et al. A Navigation Message Authentication Proposal for the Galileo Open Service. *Navigation*, 63, 2016.