# Demo: Hijacking Bluetooth's Broadcast Audio Streams using BISON

Theo Gasteiger, Carlo Alberto Boano, and Kay Römer
Institute of Technical Informatics, Graz University of Technology, Austria
{gasteiger,cboano,roemer}@tugraz.at

## Abstract

In this demo we present BISON, a novel attack on Bluetooth's broadcast isochronous streams (BISes), and demonstrate it on off-the-shelf hardware. BISON exploits the plaintext metadata used for stream synchronization as well as the vague specification of the Broadcast_Code exchange to take over ongoing BISes and manipulate their content. We showcase the feasibility of BISON by hijacking a broadcast audio stream envisaged to be deployed in public venues.

## 1 Introduction

The Bluetooth specification has recently undergone extensive updates in order to improve performance and enable new use cases. One of the most prominent additions are the so-called *isochronous channels* [8], laying the foundation for the new LE Audio standard [6]. Isochronous channels enable time-sensitive data transmission and synchronized data rendering across multiple receiver devices, which is a key feature for audio use cases such as "true wireless earbuds". Additionally, in public spaces, BISes, another feature of isochronous channels, enable unidirectional and simultaneous audio dissemination to countless devices, paving the way for a plethora of new use cases. One of those envisaged use cases is the broadcast of audio data to large collections (potentially, an unlimited number) of devices (marketed as *Auracast* [3]), enabling the creation of assisted hearing systems in public locations [10]. Indeed, the use of BISes for audio diffusion is foreseen in locations such as theaters, museums, bars, congress centers, and transport hubs [3, 4, 6]. To ensure authenticity, integrity, and confidentiality, broadcast audio streams available in public areas may be open (unencrypted) or private (encrypted). For example, an airport may provide gate announcements in the form of an unencrypted broadcast audio stream to any surrounding device. Alternatively, to avoid any manipulation of the audio stream by third parties (i.e., man-in-the-middle attacks changing the contents of the audio stream), one may encrypt the BIS using a Broadcast_Code (essentially, a passkey used to derive a symmetric session key). This encryption is especially relevant in applications such as turn-by-turn navigation for visually-impaired people or wellness tips in a medical waiting room where a manipulation of the audio information may result in injury or harm. Unfortunately, most of the envisaged applications broadcasting audio streams in public spaces favor an effortless joining procedure rather than ensuring authenticity, integrity, and confidentiality. Although it is recommended for Auracast transmitters in public spaces to use *unencrypted* broadcast audio streams, certain scenarios require the use of encrypted audio streams where the distribution of the Broadcast_Code should be "easy to do" [2] and is envisaged with out-of-band methods such as QR codes displayed at prominent locations, or NFC tags mounted in dedicated areas [1, 6]. Concretely, this means that a user can simply scan a QR code at the gate (or tap his/her earbuds against a dedicated NFC terminal) to join a private BIS and receive the encrypted gate announcements [4]. However, *anyone* can do so, including a malicious actor, who can hence easily gain access to the key material of the encrypted BIS.

In this demo, we showcase BISON, a novel attack on Bluetooth's BISes [5], by taking over an ongoing broadcast audio stream (i.e., by impersonating the source of data transmissions) and by manipulating its content (i.e., by forging arbitrary payloads). We do so by exploiting the plaintext metadata used for BIS synchronization and Bluetooth's channel map update procedure.

## 2 BISON: Attack Overview

Even though BISes are designed to be secure, malicious actors can, similar to legitimate receiver devices, synchronize to an ongoing stream and use BISON to take over the BIS and manipulate its content. The BISON attack itself is composed of three distinct phases: ❶ synchronization, ❷ overshadowing, including the manipulation of the (encrypted) payload, and ❸ forking, i.e., forcing a receiver to switch to a forged stream operating on a different channel map. Figure 1 illustrates these three distinct phases, where Alice represents an isochronous broadcaster, Bob an isochronous receiver and Mallory the malicious actor executing all three attack phases.

**Phase ❶: Synchronization.** In general, isochronous receiver nodes, such as Bob, can synchronize to ongoing BISes

by executing the periodic advertising synchronization establishment procedure followed by the broadcast isochronous synchronization establishment procedure, provided they are in communication range of the isochronous broadcaster (such as Alice). Malicious actors such as Mallory can also exploit this procedure and calculate the timing as well as the channel used in future BIS events. In other words, Mallory can synchronize to the ongoing BIS, and calculate when and on which channel Bob will listen in the future. Although BIS payloads can be encrypted, the Bluetooth core specification does not foresee encryption of the metadata used for establishment, enabling not only `BISON` but also selective jamming attacks [2].

**Phase ⓫: Overshadowing.** Mallory, knowing the future moves of Bob, can now overpower Alice (e.g., by increasing the transmission power) and, due to the capture effect [7], prevent Bob from receiving Alice's broadcast. If Mallory transmits invalid packets, Bob is considered a victim to a selective jamming attack, whereas if Mallory transmits forged packets, Bob is considered a victim to an impersonation attack. Mallory can also exploit the out-of-band key material exchange, which "should be easy to do" [2, 6], to forge the content of encrypted BIS packets. Indeed, the Auracast documentation [1] leaves it up to the implementer how such exchange should be implemented, recommending either a static value printed on a label, or a non-static value displayed on a screen/TV. Additionally, a control subevent, originally intended for channel blacklisting, can be transmitted, indicating a channel map update starting at a given instant. This control sub-event tells Bob that the ongoing BIS will use a different channel map and bypasses the limitation that an attacker needs to be in close proximity to the receiver or have the ability to send at a very high transmission power.

**Phase ⓬: Forking.** After the specified instant in the channel map update control subevent is reached, Mallory and Bob update their channel maps, allowing Mallory to diverge from the BIS broadcasted by Alice. As a result, Bob will inadvertently listen to the BIS broadcasted by Mallory, under the impression of listening to Alice without requiring further overpowering of the legitimate BIS by Mallory.

## 3 `BISON`: Demonstration

We demonstrate the effectiveness of `BISON`[1] on two Nordic Semiconductor nRF5340 audio development kits, representing Alice and Bob, and one nRF52840 development kit representing Mallory. All three boards utilize the Zephyr real-time operating system [9], implementing an envisaged *Auracast* scenario, where Alice acts as a broadcast audio transmitter and Bob acts as a broadcast audio receiver. Mallory uses a modified implementation of the *synchronized receiver* sample, executing all phases of the `BISON` attack. This setup enables the audience to take the role of Mallory and, via one button press, hijack the ongoing unencrypted[2] broadcast audio stream, injecting malicious audio data. Due to the channel map update executed in the last attack phase, Mallory (i.e., the audience) is also able to increase the distance to Bob, observing the successful impersonation attack even though Alice is still in close proximity to Bob. Moreover, the audience can validate `BISON` by listening to the audio stream itself. Before the exploit is executed, Alice is broadcasting voice announcements, which get replaced with music during the execution of the attack. Additionally, a separate screen shows the inner workings of `BISON` in detail, providing insight into the individual attack phases and the link quality during execution. Finally, RGB LEDs on the two audio development kits can be observed, representing the current channel map, encoded in a specific color.

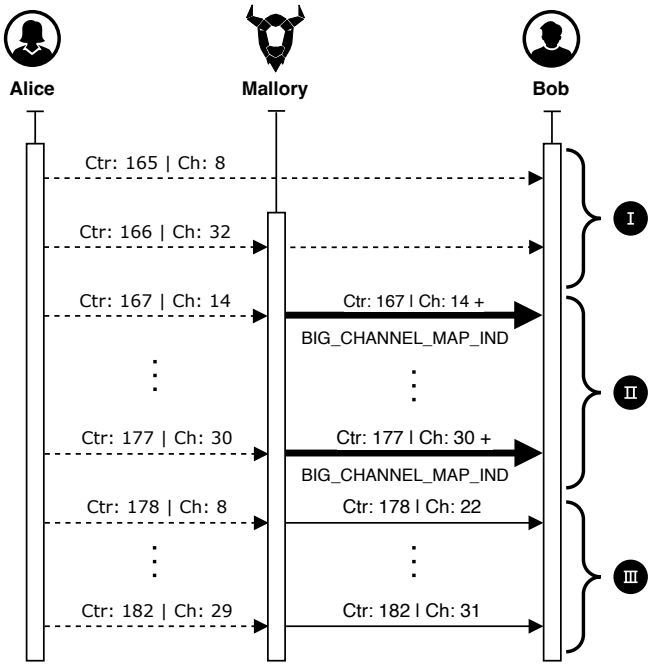


**Figure 1. `BISON` architecture.** Mallory synchronizes to the ongoing BIS, broadcasted by Alice, in attack phase I. This synchronization is followed by the overpowering of the BIS and the addition of a channel map update subevent in phase II. As soon as the specified instant is reached, Mallory and Bob use the updated channel map, resulting in a complete divergence from the BIS broadcasted by Alice in phase III.

## 4 References


[1] Bluetooth SIG. Auracast Simple Transmitter Best Practices Guide, v1. Oct 2022.

[2] Bluetooth SIG. Bluetooth Core Specification, v5.4. Jan 2023.

[3] Bluetooth SIG. Key Use Cases in Public Locations. https://www.bluetooth.com/auracast/public-locations. Last access: Mar 2023.

[4] S. Cohen. Auracast looks to radically change how you use Bluetooth. https://bit.ly/3ziihQ3. Last access: Mar 2023.

[5] T. Gasteiger et al. BISON: Attacking Bluetooth's Broadcast Isochronous Stream. In *Proc. of the 20th EWSN Conf.*, 2023.

[6] N. Hunn. *Introducing Bluetooth LE Audio*. 2022.

[7] K. Leentvaar and J. Flint. The Capture Effect in FM Receivers. *IEEE Transactions on Communications*, 24(5), 1976.

[8] K. Ren. 10 Frequently Asked Questions on LE Isochronous Channels. https://bit.ly/40kDDbz, 2020.

[9] Zephyr Project. About the Zephyr Project. https://zephyrproject.org/learn-about. Last access: Mar 2023.

[10] A. Zignani. LE Audio, Auracast Broadcast Audio, and the Future of Bluetooth Audio. https://bit.ly/3lSxIvg. Last access: Mar 2023.


[1] https://iti.tugraz.at/bison

[2] Due to considerations of practicality and convenience with respect to comprehensible packet content visualization.