Cut the Cord: Forensic Evidence by Exploiting Wireless Vulnerabilities

Ahmad Shakleya, Ruben Nietvelt, Thomas Janssen
Faculty of Applied Engineering, imec-IDLab, University of Antwerp, Antwerp, Belgium
{ahmad.shakleya@student., ruben.nietvelt@, thomas.janssen@}uantwerpen.be

***Abstract*—Cable-bound forensic extraction is frequently blocked by devices that are locked, encrypted, or physically damaged. However, they can still communicate over short-range wireless networks, for example, Bluetooth and Wi-Fi, which offer other points of entry. This study investigates the added forensic value of recent wireless vulnerabilities. The MITRE Common Vulnerabilities and Exposures (CVE) database was used to collect 19 CVEs with public proof-of-concept (PoCs). Three relevant Bluetooth exploits were identified using a two-axis Usability × Feasibility matrix: CVE-2022-30144, CVE-2023-45866, and CVE-2023-24871. Results indicate that a structured scoring method aids investigators in prioritising CVEs for forensic value, and that wireless exploits can produce court-relevant evidence without physical access.**

***Index Terms*—forensics, cybersecurity, CVE, wireless communication technologies**

I. Introduction

Cable-based data extraction often fails on physically damaged or inaccessible devices, and chip-off recovery is delicate, time-consuming, and frequently results in unreadable dumps due to hardware encryption. This study explores whether recent wireless vulnerabilities can be exploited to obtain forensic artefacts such as logs, chat databases and volatile keys without physical access, and proposes a forensically sound, ethical methodology to help law-enforcement perform faster, less intrusive investigations.

As new wireless vulnerabilities continue to emerge, we posed the following overarching question: What is the added forensic value of recently discovered wireless vulnerabilities? The rest of this study is guided by the three research questions we propose. RQ1: What are the publicly known wireless exploits that were revealed for mobile or desktop devices between 2022 and 2024? RQ2: Before allocating laboratory resources, how can these exploits be ranked according to their expected feasibility and usability? RQ3: Once replicated in the lab, what is the concrete forensic return on investment of these exploits for digital investigations carried out by a law-enforcement agency?

II. Related Work

Bernardo et al. [1] suggested a Design-Science toolkit that organises evidence discovery, acquisition, analysis, and reporting. Their method mainly focuses on conventional, cable-bound forensic acquisition, which is ineffective when devices are physically damaged or unreachable. This is the exact forensic gap that our work attempts to fill through wireless exploitation.

Modern laptops, smartphones, and IoT devices almost always include Bluetooth or Wi-Fi, and because these radios are enabled by default and use complex protocols, they significantly expand the devices' attack surface. Beck et al. [2] classify wireless threats such as RFID, ZigBee, Qi and Li-Fi into active attacks involving signal modification, i.e. spoofing and jamming, and passive attacks, based on eavesdropping or traffic analysis. We mainly concentrate on active attacks since they have the most potential for forensic value because they can gain system-level access to devices that would otherwise be unreachable. The rest of this section therefore focuses on Bluetooth and Wi-Fi vulnerabilities and on the forensic frameworks that take benefit from them.

Bluetooth vulnerabilities can occur at different layers of the protocol. At the link layer, reusing session keys during the setup of a Bluetooth session violates both forward and future secrecy, as demonstrated by Antonioli's BLUFFS attack [3]. Six unique vulnerabilities were reproduced on Qualcomm, Apple, and Intel chipsets, allowing device spoofing and decryption of previous communications. At the application layer, vulnerabilities occur higher in the software stack. Such exploits are not mitigated by transport-layer patches.

Even without jailbreaking an iPhone, Bluetooth information can be useful. An SQLite Bluetooth Low Energy (BLE) database was extracted from an iPhone 13 by Fedynyshyn et al. [4] to illustrate this. This database contains detailed logs of nearby Bluetooth devices that were detected, but were not paired or connected. By using the database's `LastSeenTime` field, proximity events on locked devices could be reconstructed. This enables forensic investigators to establish timelines, infer associations between individuals, and detect suspicious behavioral patterns based on timestamped Bluetooth logs.

In their four-phase penetration-testing process for IEEE 802.11 networks, Akhtar and Rawol [5] demonstrated how legacy WEP, WPA, and WPS are still compromised by tools such as `Airodump-ng`, `Aircrack-ng`, and `Reaver`; evil-twin access points further enable credential phishing. Although these standards have long been superseded by WPA2 and WPA3, forensic investigators still encounter them in the field due to legacy hardware, compatibility, or transition modes.

III. Methodology

We reduced a large number of recently disclosed wireless vulnerabilities using a 3-stage process. The three stages are described below. They parallel the Design-Science Research (DSR) framework of Bernardo et al. [1]: Stages 1–2 cover device discovery and tool selection, while Stage 3 runs the full DSR cycle for the selected wireless exploits.

A. Stage 1 - CVE Discovery

a) Data sources and query strategy: We cloned the Trickest GitHub repository [6], which is a mirror of MITRE Common Vulnerabilities and Exposures (CVE) entries that are automatically generated and contain publicly available proof-of-concept (PoC). We used the snapshot and limited the study window to end in 2024 because there were no 2025 PoCs available at the time of cloning. After that, we executed a Python filter script that loops over the repository's annual folders and returns markdown files with all of the provided keywords in their text. "Bluetooth, iOS," "Bluetooth, Windows," "Bluetooth, Android," "Bluetooth, automotive," "Wi-Fi, iOS," "Wi-Fi, Windows," "Wi-Fi, Android," and "Wi-Fi, automotive" were the eight keyword pairs that were filtered upon. Windows/iOS/Android CVEs were limited to 2022–2024 because mainstream operating systems (OSs) receive regular updates and older exploits are typically patched; automotive CVEs were accepted from 2018–2024 because tooling for earlier entries is either unavailable or outdated. This search yielded in 19 distinct CVEs that satisfied these initial criteria.

B. Stage 2 - CVE Prioritisation and Planning

Because every selected exploit must be replicated, benchmarked, and forensically validated, the study prioritises depth over breadth. We therefore use a selection method that trades forensic value (i.e., amount and significance of evidence) against engineering cost (i.e., effort and specific equipment needed).

The criteria were derived from Common Vulnerability Scoring System (CVSS) score families [7]. In particular, we used Temporal metrics i.e. exploit-code maturity, remediation level and report confidence, and Base metrics such as attack complexity and required privileges. We refined the criteria with law enforcement forensic engineers: factors that affect lab effort form the Feasibility axis, while those that affect evidence value form the Usability axis.

Weights for the four usability (U) and feasibility (F) criteria were selected so that the sum of the values on each axis equals ten. We started by assigning reliability & predictability (F) and forensic usability (U) the highest weight (4) because these two factors determine whether an exploit is useful in practice. Low forensic usability has little investigative value, and frequent crashes or uninterpretable traces merely waste analyst time. Only exploits that overwrite the data could truly compromise evidence, and those are filtered out later in Stage 3. To assign a provisional reliability score before any lab work, we reviewed community feedback already embedded in the exploit itself, namely the success-rate notes in the PoC's README and stability discussions in its GitHub Issues page. Should subsequent validation still reveal frequent crashes or useless artefacts, the score is downgraded, reflecting the risk that such instability could contaminate evidence. Exploitability was assigned a weight of three. An off-the-shelf PoC accelerates laboratory work, but it does not increase the factual value on its own, in contrast to the first two criteria. The "effort" criteria-target surface (U), technical feasibility (F), exploit requirements (F), and attack prerequisites (F) were given weights of two. Each of them influences the amount of expertise, time, additional hardware, or other data that the analyst needs to have, but once the more important evidentiary criteria are met, none of them matter. Because an officially patched vulnerability frequently remains accessible on field devices that have never been updated, absence of patches was finally given the lowest weight of one.

TABLE I: Usability criteria (weights sum to 10) with forensic rationale.

Criteria	Weight	Scoring description	Forensic rationale
Exploitability	3	0: No known exploit 1: PoC exists but not operational 2: PoC works after minor modification 3: Fully weaponised exploit (e.g. Metasploit)	Presence of an off-the-shelf exploit speeds lab replication and shortens time to evidence.
Target surface	2	0: Rarely used software 1: Widely used platform (Windows, Linux, Android) 2: Cross-platform	Broader platform coverage increases the likelihood an exploit applies to devices encountered in the field.
Patch absence	1	0: Already patched 1: No patch exists	Unpatched vulnerabilities are more likely to remain exploitable on real devices, improving future utility.
Forensic usability	4	0: No forensic relevance 2: Limited relevance (e.g. last-dialled number) 4: High relevance (e.g. useful timestamps, logs, network traceability)	Measures the evidence value for investigations (highest weight because end goal is actionable forensic data).

Plotting these variables as orthogonal axes (i.e., Usability × Feasibility) allows us to clearly see the trade-offs and identify both "quick wins" and longer shots with greater impact. We selected three high-priority candidates for comprehensive laboratory validation from the original CVE collection using this grid. We aimed for the top right corner in our CVE matrix (see Figure 1), because those CVEs were highly useful and low in complexity. The full scoring sheet, containing all 19 CVEs, is presented in the results in Section IV. The chosen CVEs are listed in Table III.

C. Stage 3 - Exploitation and Documentation

An HP 14-cf laptop running Ubuntu 24.04 LTS (Intel i5-1035G1) served as the attacking host for all of the experiments.

TABLE II: Feasibility criteria (weights sum to 10).

Criteria	**Weight**	**Scoring description**	**Reproducibility rationale**
Technical feasibility	2	0: Requires deep OS/hardware knowledge (advanced bypass) 1: General exploitation knowledge 2: Basic skills (run script/PoC out of the box)	Reflects the analyst skill curve: lower scores indicate complex, specialist work and higher setup cost.
Exploit requirement	2	0: Chain of multiple vulnerabilities 1: Single exploit with external hardware 2: Single exploit without external hardware	Penalises chained exploits or those needing rare hardware, which increase setup time, cost and maintenance burden.
Attack prerequisites	2	0: Needs specific conditions or credentials (device unlocked) 1: Some requirements (e.g. 4-digit PIN) 2: No special conditions needed	Rewards standalone attacks that work on locked/unowned devices and do not require prior credentials.
Reliability & predictability	4	0: Unstable 2: Some stability issues (race conditions) 4: Highly reliable	Given the risk of evidence contamination and maintainability concerns, stability receives the highest weight.

1) CVE-2022-30144 - Wireless Keystroke Injection (Windows 10/11): A Nordic nRF52840 dongle [8] was connected to the Ubuntu host after being flashed with Layakk's custom BLE-keyboard firmware [9]. Previously paired with its owner's wireless keyboard, the target was a Dell Latitude 5xxx laptop (Windows 10 22H2, Bluetooth driver 22.90.0.5) that was locked at the Windows logon screen.

To identify the keyboard's BLE address, we performed passive sniffing. This also revealed the Generic Attribute Profile (GATT) handle (the Human Interface Device (HID) characteristic identifier) and the report size [10]. The GATT handle ((65 535 possibilities) and report size (4 options) are not broadcast and generally must be brute-forced, while opcode 0x1B, which stands for 'Handle Value Notification", is the only fixed value for BLE keyboards [11]. We paired the keyboard to retrieve its GATT handle and report size, copied those values into the dongle, and injected a payload that opened Notepad, typed text, and saved it to confirm the exploit. In an actual forensic scenario the investigator will not have the keyboard, so the handle/size must be brute-forced.

2) CVE-2023-45866 - BlueDucky HID: Although CVE-2023-45866 is cross-platform, we validated it on readily available Android test devices; BlueDucky 2.1 ran on the attacker host without additional hardware [12]. Two devices were tested: the Samsung J7 DUOS Pro (Android 7.0) and the Samsung Galaxy Tab A9+ (Android 14, Dec-2023 patch missing). Following the acquisition of each device's Bluetooth Media Access Control (MAC) address, the framework introduced HID keycodes that the target accepted, simulating the pairing of a wireless keyboard without the need for any credentials.

3) CVE-2023-24871 - Extended-Advertising Remote Code Execution / Local Privilege Escalation (Windows 10/11): A CSR µEnergy Bluetooth 4.2 dongle [13] was attached to a Dell Latitude 5xxx (Windows 10 22H2). We used check_vuln.bat, which is available on the GitHub repository ynwarcs/CVE-2023-24871 [14], to determine the maximum advertising-frame size; a value of at least 514 bytes is required for the target to be vulnerable. To establish a baseline, we attached the dongle to a Windows 11 (22H2) virtual machine (VM), the most recent release still marked as vulnerable [15]. We then ran capability checks on the Windows 10 host and vulnerability checks on the VM.

IV. Results

A. *Stage 1 - CVE Discovery*

The keyword and date filters described in Section III-A returned 19 distinct CVEs that have a PoC already available:

The score matrix presented in Fig. 1 is derived from this search. The remaining 16 CVEs offer a ready-made backlog for future forensic research, even if the current analysis only replicated the top three Bluetooth CVEs.

B. *Stage 2 - CVE Prioritisation and Planning*

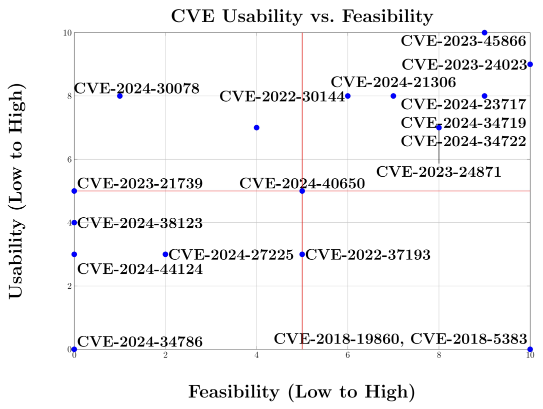

Fig. 1: Usability–Feasibility plot for the 19 CVEs. Scores reflect expected value if preconditions hold.

C. *Stage 3 - Exploit Evaluation and Documentation*

1) CVE-2022-30144 - Wireless Keystroke Injection: The keyboard's GATT handle and report size remained unknown (a read returned the default 0x0000), meaning that full replication would require brute-forcing 262 140 combinations. Therefore, we hard-coded one handle/size pair and focused on benchmarking the PoC's transmission component displayed in Table IV. This is still helpful due to the independent wall-clock timing of the PoC: It is one-way communication.

TABLE III: Priority scores of the selected CVEs based on the criteria in Table I and Table II (U = Usability, F = Feasibility). Although other CVEs (CVE-2024-23717, CVE-2024-34719, and related Android exploits) received slightly higher scores, we chose these CVEs because they provided a variety of desktop and mobile targets, and match the available Bluetooth hardware.

CVE	Target OS	Explanation	Goal
CVE-2022-30144	Win 10/11	High overall score and compatible with available hardware (nRF52840 dongle + target laptop).	Spawn an elevated command prompt to use as a data-collection bridge-head.
CVE-2023-24871	Win 11	Combines service-level RCE with a reliable LPE chain; no external dongle needed on the host VM.	Obtain an administrator shell on a locked system; Combines service-level RCE with a reliable LPE chain and does not require an external dongle on the host VM.
CVE-2023-45866	Android $\leq$ 14	Cross-platform HID RCE, zero pairing requirement, and the host already has a built-in Bluetooth controller.	Link the analyst device, enable Wireless ADB / Developer Options, and pull a full filesystem dump from a device with a broken USB port.

TABLE IV: Break-down of the CVE-2022-30144 attack script (mean of 100 runs). The validation of the forensic value is non-existing, because the exploit did not reach the target device.

Timing chunk	Avg. (s)	Operation summary
load_config	0.00011	Parse config.json.
tokenize_script	0.00003	Split payload into high-level tokens.
generate_reports	0.00075	Convert tokens to 11-byte HID reports.
init_device	0.03300	Reset HCI socket.
config_adv	0.00630	Craft & enable a custom advertisement that masquerades as the wireless keyboard.
config_exploit	0.27819	Connect to the victim, perform ATT queries.
start_exploit	0.00293	Transmit HID reports.
TOTAL	**0.32727**	End-to-end runtime.

2) CVE-2023-45866 - BlueDucky HID: This vulnerability has been validated and works as intended. Figure 2 summarises the field workflow. BlueDucky first asks for a target MAC; if omitted, it performs a passive scan and stores hits in known_devices.txt. The unauthenticated HID connection is attempted only if Bluetooth is enabled and the handset advertises HID.

Patch level determines the next hop:

- Android $\leq$ 10: Exploit works out-of-the-box. Forensic collection must be collected manually because Wireless ADBs is unavailable.
- Android 11-14: If the Dec-2023 security patch is not installed, the full script runs; otherwise, the attack fails unless the handset was previously paired.
- Android $\geq$ 15: Vulnerability is already patched. This CVE cannot be used for that.

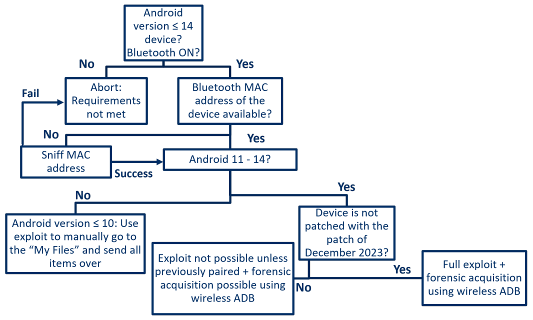
Android version ≤ 14 device? Bluetooth ON?
No
Yes
Fail
Abort: Requirements not met
Bluetooth MAC address of the device available?
No
Yes
Sniff MAC address
Success
Android 11 - 14?
No
Yes
Android version ≤ 10: Use exploit to manually go to the "My Files" and send all items over
Device is not patched with the patch of December 2023?
Exploit not possible unless previously paired + forensic acquisition possible using wireless ADB
No
Yes
Full exploit + forensic acquisition using wireless ADB

Fig. 2: Attack Tree of CVE-2023-45866 for Android OS. This CVE has been validated and is ready for use in further forensic research.

On the Galaxy Tab A9+, the payload awakened the device, used a presumed PIN to unlock it, enabled the Developer Options/Wireless Android Debug Bridge (ADB), connected the device to a rogue Wi-Fi access point (AP), paired over ADB, and pulled a full filesystem dump since the device was rooted; on the Samsung J7 DUOS Pro (Android 7) Wireless ADB was not available [16], so the script unlocked the screen, opened "My Files," and sent each folder to the examiner's host via Bluetooth.

3) CVE-2023-24871 - Extended-Advertising RCE / LPE: Both physical and virtual test beds failed the prerequisite for this exploit:

- Dell Latitude 5xxx (physical target): maximum advertising-packet length was 31 bytes - typical for Bluetooth 4.2 controllers (see [17]) and far below the 514 bytes required by the PoC.
- Windows 11 Insider VM: Lacked direct Host Controller Interface (HCI) passthrough. Attaching a CSR µEnergy® BLE USB dongle merely reproduced the same 31-byte limit because the device is also Bluetooth 4.2 [17].

V. Discussion

We evaluated the utility of our two-axis prioritization model (RQ2). Plotting forensic usability against engineering feasibility produced a clear filter (Fig. 1). Three CVEs (from the 19 CVEs in the matrix) in the upper right quadrant were replicated, while lower left candidates were put aside for further research. Although three Android-only CVEs (CVE-2024-23717 [18], CVE-2024-34719 [19] and CVE-2024-34722 [20]) scored slightly higher than the final selections, they were excluded for practical reasons. A fourth candidate, CVE-2024-21306 [21], was left for future work.

Thirdly, we assessed the added forensic value of the three reproduced exploits (RQ3), outlining their concrete return on investment for investigators.

- CVE-2022-30144: The CVE lets an attacker bypass a locked Windows laptop by replaying HID keystrokes to spawn an elevated command shell and open a reverse secure shell for evidence extraction. The practical challenge is brute-forcing the unknown handle/size pair (262,140 combinations at 0.32 s per attempt).

- CVE-2023-45866: The exploit allows remote access to an Android device over Wi-Fi even if its USB port is broken. By enabling Wireless ADB, investigators can pull a full filesystem dump to the examiner, yielding results comparable to cable-based tools. If ADB is not available, user-data can be transferred without a pairing request.
- CVE-2023-24871: Reproduction failed because our test hardware did not support the 514-byte extended advertising frames; investigators should first verify that the target (or a same-model test device) has a Bluetooth 5.0+ radio with extended-advertising enabled before attempting this exploit.

The PoC for CVE-2022-30144 and CVE-2023-24871 failed in our experiments; however, their high matrix scores reflect their conditional forensic utility (value when preconditions hold) rather than guaranteed success on every hardware configuration. All measurements were taken on a single host (HP 14-cf, Intel i5-1035G1), so timings and exploit stability may differ on other systems or Bluetooth adapters. Additionally, CVE-2023-45866 is limited to Linux systems; CVE-2024-21306 [21], a similar vulnerability affecting Windows, should be evaluated in future work.

VI. Conclusion

Recently disclosed wireless exploits can, if viable, yield court-admissible evidence without needing physical access. Our step-by-step method and two-axis scoring matrix enable researchers to reproduce exploits or assess new CVEs and to decide systematically when a wireless ‘cut-the-cord” extraction is preferable to chip-off.

a) Added forensic value of validated CVEs: In particular, this study showed that CVE-2023-45866 offers added forensic value by enabling full filesystem extraction from Android devices, even in cases where the USB connection is totally destroyed. CVE-2022-30144, on the other hand, emphasised the necessity for additional optimisation prior to deployment, pointing out practical constraints due to the brute-force effort needed. Last but not least, CVE-2023-24871’s unsuccessful validation provided specific hardware requirements, acting as an important boundary condition that helps forensic investigators quickly determine the viability of an attack.

b) Future work: The next milestone is cross-platform support for CVE-2023-45866, which would allow investigators to screen mixed device groups from a single toolchain by transferring the HID-injection code to Windows, iOS/iPadOS, and Linux. Additionally, CVE-2023-24871 should be re-tested on Bluetooth 5.0 devices that support extended advertising longer than 512 bytes. If the exploitation is successful, investigators will have access to a forensic RCE–LPE pathway for Windows computers.

References

[1] B. M. V. Bernardo, H. S. Mamede, J. M. P. Barroso, and V. M. P. D. dos Santos, “Mobile device forensics framework: A toolbox to support and enhance this process,” *Emerging Science Journal*, vol. 8, no. 3, pp. 972–998, 2024.

[2] S. Beck, M. Raavi, C. Dale, K. Weishalla, and B. Worrell, “Survey of side-channel vulnerabilities for short-range wireless communication technologies,” in *Proc. IEEE Int. Conf. Electro Information Technology (eIT)*, 2024, pp. 450–456.

[3] D. Antonioli, “BLUFFS: Bluetooth forward and future secrecy attacks and defenses,” in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS ’23)*, 2023, pp. 636–650. [Online]. Available: https://doi.org/10.1145/3576915.3623066

[4] T. Fedynyshyn, I. Opirskyy, and O. Mykhaylova, “A method to detect suspicious individuals through mobile device data,” in *Proc. 2023 IEEE 5th Int. Conf. Advanced Information and Communication Technologies (AICT)*, 2023, pp. 82–86.

[5] Z. B. Akhtar and A. T. Rawol, “Uncovering cybersecurity vulnerabilities: A Kali Linux investigative exploration perspective,” *International Journal of Advanced Network, Monitoring and Controls*, vol. 9, no. 2, pp. 11–22, 2024. [Online]. Available: https://doi.org/10.2478/ijanmc-2024-0012

[6] Trickest, “cve (GitHub repository),” Online, accessed: Feb. 24, 2025. [Online]. Available: https://github.com/trickest/cve/tree/5d308336fbdbfda60aa94d167654f35456474ec1

[7] T. Olaes, “Understanding CVSS temporal metrics: How they affect risk,” Online, accessed: May 20, 2025. [Online]. Available: https://www.balbix.com/insights/temporal-cvss-scores/

[8] N. Semiconductor, “nRF52840 dongle,” Online, accessed: May 1, 2025. [Online]. Available: https://www.nordicsemi.com/Products/Development-hardware/nRF52840-Dongle/

[9] Layakk, “WKI (Wireless Keystroke Injection),” Online, accessed: Apr. 16, 2025. [Online]. Available: https://github.com/Layakk/WKI

[10] T. Instruments, “Generic attribute profile (GATT) in the BLE5-stack user guide,” Online, accessed: May 1, 2025. [Online]. Available: https://software-dl.ti.com/lprf/simplelink_cc2640r2_sdk/1.35.00.33/exports/docs/ble5stack/ble_user_guide/html/ble-stack/gatt.html

[11] ——, “TI BLE5-stack API documentation: ATT methods,” Online, accessed: May 1, 2025. [Online]. Available: https://software-dl.ti.com/simplelink/esd/simplelink_cc13x2_26x2_sdk/4.30.00.54/exports/docs/ble5stack/ble_user_guide/doxygen/ble/html/group__att__method__defines.html

[12] pentestfunctions, “Blueducky: CVE-2023-45866 implementation (Using DuckyScript)—unauthenticated peering leading to code execution (Using HID keyboard),” Online, accessed: May 12, 2025. [Online]. Available: https://github.com/pentestfunctions/BlueDucky

[13] Qualcomm, “CSRA64215 Bluetooth 4.2 Bluetooth Low Energy chipset,” Online, accessed: May 1, 2025. [Online]. Available: https://www.qualcomm.com/products/internet-of-things/consumer/audio/csra64215

[14] ynwarcs, “CVE-2023-24871: Pocs & exploit,” Online, accessed: May 12, 2025. [Online]. Available: https://github.com/ynwarcs/CVE-2023-24871

[15] N. I. of Standards and Technology, “CVE-2023-24871—national vulnerability database (NVD),” Online, accessed: May 1, 2025. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2023-24871

[16] A. Developers, “Android debug bridge (adb),” Online, accessed: May 1, 2025. [Online]. Available: https://developer.android.com/tools/adb#connect-to-a-device-over-wi-fi

[17] S. Labs, “Extended advertising example: Advertising, scanning; Bluetooth LE v2.13,” Online, accessed: May 13, 2025. [Online]. Available: https://docs.silabs.com/bluetooth/2.13/bluetooth-code-examples-stack-features-adv-and-scanning/extended-adv-example

[18] N. I. of Standards and Technology, “CVE-2024-23717—national vulnerability database (NVD),” Online, accessed: May 20, 2025. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2024-23717

[19] ——, “CVE-2024-34719—national vulnerability database (NVD),” Online, accessed: May 20, 2025. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2024-34719

[20] ——, “CVE-2024-34722—national vulnerability database (NVD),” Online, accessed: May 20, 2025. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2024-34722

[21] ——, “CVE-2024-21306—national vulnerability database (NVD),” Online, accessed: May 20, 2025. [Online]. Available: https://nvd.nist.gov/vuln/detail/CVE-2024-21306