International Journal of Emerging Trends in Computer Science and Information Technology
ISSN: 3050-9246 | https://doi.org/10.63282/3050-9246.IJETCSIT-V1I1P111
Eureka Vision Publication | Volume 1, Issue 1, 95-116, 2020

Original Article

Bluetooth Interoperability across Heterogeneous Handset and Accessory Ecosystems: A Test Framework

Krishnan Visweswaran
Independent Researcher, USA.

Abstract - Bluetooth is the default short-range radio for consumer handsets and their accessories, yet interoperability failures persist between devices that have passed conformance certification. This article tackles the issue of systematically assessing Bluetooth interoperability in a heterogeneous handset and accessory environment. It synthesizes evidence from the Bluetooth Core Specification, profile specifications, qualification documentation, platform vendor documentation, and peer-reviewed research on Bluetooth security and link performance. It provides a four-level taxonomy that distinguishes protocol conformance, profile interoperability, end-to-end functional compatibility and user-perceived reliability, and outlines evidence needed for each level. It proposes a thirteen-component framework comprising fourteen test categories, each specified by objective, preconditions, procedure, expected outcome, failure conditions, required logs and evaluation metric. It defines eight formal metrics, with units of measurement, denominators, observation windows, and interpretation rules specified, and an explicit policy for missing data, unsupported features, timeouts and invalid runs. It provides a source-based traceability analysis that maps out the documented failures in interoperability and in security and links them to the tests that would have caught them, and suggests a test-coverage matrix. No new apparatus measurements were made; all quantitative figures quoted are as given by the source quoted for the conditions the source reports. The framework is offered as a reproducible validation protocol for interoperability laboratories, handset vendors, accessory developers and quality-assurance teams.

Keywords - Bluetooth Interoperability, Protocol Conformance, Bluetooth Low Energy, Test Automation, Device Compatibility, Wireless Testing, Interoperability Assessment, Cross-Platform Validation, Bluetooth Security.

1. Introduction

Bluetooth is a widely used short-range wireless technology in consumer handsets and accessories. According to data from the Bluetooth SIG, over 4.0 billion Bluetooth devices were shipped in 2020, of which 152 million were earbuds [21]. Handsets interact with headsets, speakers, wearables, input devices, and vehicle systems, which are often designed, implemented, qualified, and updated independently. From the user's perspective, connectivity depends not only on individual-device conformance but also on whether the pair can discover one another, negotiate compatible features, establish the required security, exchange data, and recover from connection failures.

Conformance does not guarantee interoperability between independently developed devices. Bluetooth qualification assesses whether a design meets the applicable qualification requirements, while interoperability testing evaluates interactions between implementations, including cases not fully covered by specification-based conformance tests [3]. Conformance evidence demonstrates correct behaviour against a reference test system but does not guarantee interoperability with every peer, particularly when optional features, timing, and error handling differ. Key negotiation was shown to be manipulable so that encryption key entropy could be forced as low as one byte, with all seventeen chips tested vulnerable [5], [6]. In the authentication part of the secure connection establishment process, it is demonstrated that it is possible to impersonate a bonded peer without the long term key, impacting 30 out of 31 devices [7]. Malformed over-the-air packets were shown to deadlock, crash or corrupt the memory of Bluetooth Low Energy (LE) system-on-chip implementations, yielding eleven new vulnerabilities and thirteen CVE identifiers across twelve devices from eight vendors [8], [9]. These findings concern different failure classes and should not all be characterized as interoperability defects; however, they demonstrate the need to assess functional compatibility, robustness, and security separately.

It is a problem of technique. Qualification employs a profile test system through a graphical environment [15]; some interoperability cases were suggested but not required [3]; different interfaces, bonding policies and background-execution constraints are exposed by the platform vendors [16]–[18]; and a variety of devices are exercised, repetitions are performed, and metrics are taken during laboratory practice. Two different labs can then determine coverage for different handset-accessory combinations and still come to the right answer, since neither has explicitly stated a coverage model, a denominator or an observation window. Existing work addresses isolated facets discovery latency [13], timeliness under interference [14], protocol-level attacks [5], [7], [8], [11], [12], and profile-test automation [15] but no consolidated framework treats the handset–accessory pair as the unit under test across the whole connection life cycle.

The goals of the article are: To build a consistent evidence base of causes of Bluetooth interoperability; To differentiate between interoperability level claims; To identify an implementable test framework; To define reproducible metrics; and To implement traceability from documented failure to test coverage. They are related to five research questions. RQ1: which device, platform, firmware, profile and environmental dimensions determine whether a handset–accessory pair interoperates? RQ2: How to separate protocol conformance, profile interoperability, end-to-end functional compatibility, and user perceived reliability and what evidence do they need? RQ3: what categories, preconditions, procedures and metrics constitute a complete framework? RQ4: How to relate the failures documented to test categories? RQ5: What should be done with unsupported features, missing data, timeouts and invalid runs in order for the success rates to be interpretable? The contributions are C1, a source-verified synthesis of interoperability determinants; C2, a four-level interoperability taxonomy with an explicit evidence requirement per level; C3, a thirteen-component framework comprising fourteen test categories; C4, mathematical definitions of eight metrics with denominators and interpretation rules, plus a handling policy for missing data and invalid runs; and C5, a traceability analysis mapping documented failures to detecting categories. The scope is limited to the 2020 technology baseline that was adopted at the Core Specification revision in December 2019 and the platform documentation at that time [1], [2], [16]–[18] with post-2020 sources used as labelled context. An experimental campaign was not performed and the framework is a proposed validation methodology.

2. Background and Related Work

2.1. Architecture, Versions and the Heterogeneous Ecosystem

There are two radio technologies over common host architecture, defined by Bluetooth. Basic Rate / Enhanced Data Rate (BR/EDR), also known as Bluetooth Classic, provides connection oriented services for audio streaming, telephony and serial style profiles; Bluetooth Low Energy (LE) is separately defined to provide low power and intermittent data services for advertising, scanning, and brief connection events. Both systems use host and controller components connected through the Host Controller Interface (HCI), but their upper-layer protocols differ. BR/EDR commonly uses the Service Discovery Protocol (SDP), whereas LE uses the Attribute Protocol (ATT) and Generic Attribute Profile (GATT) for service and characteristic discovery [17]. The advertised version is not the only factor that determines capability, it is a negotiated feature set of a connection. Version 5.2 (December 2019) introduced the Enhanced Attribute Protocol (EATT), which enables concurrent transactions, L2CAP packet interleaving and ATT maximum transmission unit (MTU) changes during a connection; LE Power Control, which helps a receiver to track signal strength and request power-level changes from the peer; and Isochronous Channels, which deliver time-bound data for time-synchronised processing [1], [2]. EATT is restricted to encrypted connections [2]. A handset with version 5.2 and an accessory with version 4.0 will not take advantage of these features, and it is not a defect when those features are not present. In the appropriate profile family, multiple generations are present: versions prior to the 2020 baseline (A2DP 1.2 and 1.3.2, AVRCP 1.5 and 1.6, HFP 1.7.2 and 1.8, and HSP 1.2 and later revisions) [4]. A profile version can be used to require and specify optional features, so two devices with the same profile may implement different subsets of the optional behaviour.

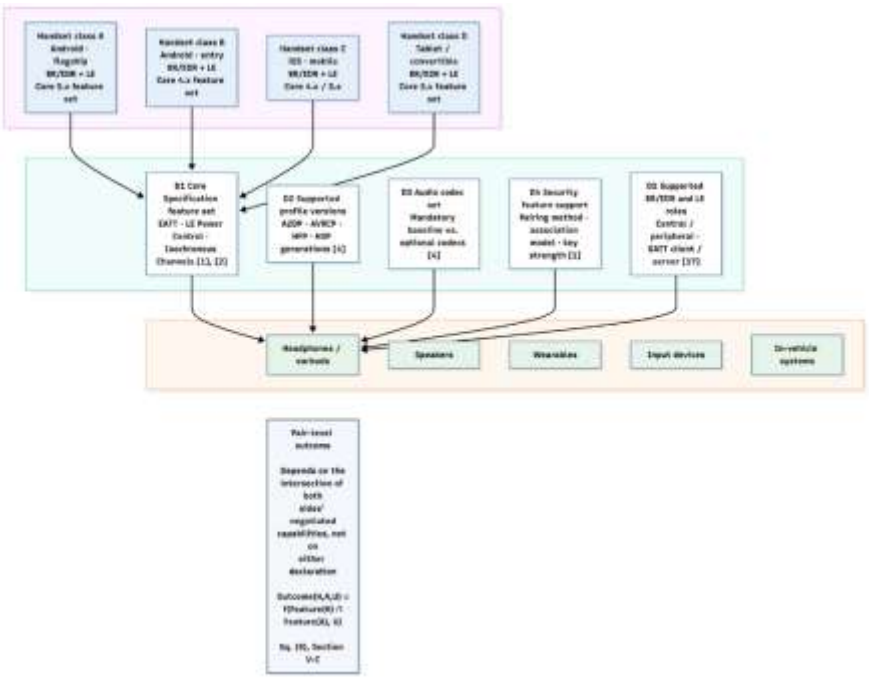


Figure 1. Heterogeneous Handset–Accessory Compatibility Ecosystem, Showing the Compatibility Dimensions That Determine Whether a Pair Interoperates

2.2. Pairing, Bonding and Security Behaviour

Pairing establishes security keys, while bonding stores the relevant keys for reuse in subsequent connections, subject to implementation and policy. The specification highlights the difference between legacy pairing and Secure Connections for BR/EDR as well as LE Legacy pairing and LE Secure Connections for LE. Association types are Just Works, Passkey Entry, Numeric Comparison and out-of-band exchange, and the model that is available on a link depends on both devices' input and output types and on the type of security features they are able to support; bonding is the placement of a link key in a key hierarchy from which subsequent session keys are generated [1]. A pairing of two capable devices can become Just Works if one peer is not capable of any display, or if one peer is not capable of any keypad. This negotiation turns documented protocol weaknesses into interoperability failures. The KNOB attack exploits in-band agreement of negotiated key entropy, allowing an attacker to force it as low as one byte; seventeen chips from vendors including Broadcom, Apple, Intel and CSR were tested and all were vulnerable, assuming the attacker is in range and able to intercept and manipulate traffic [5], [6]. The BIAS attacks target the authentication phase of secure connection establishment, covering both legacy and secure procedures; of 31 devices evaluated, comprising 28 unique chips, all but one were vulnerable, assuming the attacker can eavesdrop, manipulate and jam, and has firmware access and a baseband transceiver [7]. The Method Confusion Attack showed that two devices can be made to pair using different methods without either detecting the mismatch, establishing a man-in-the-middle position, demonstrated on off-the-shelf devices including an OnePlus 7 Pro with Android 9.0, an iPhone 11 with iOS 13.4.1, a ThinkPad W540 with Windows 10, a Samsung Galaxy Watch 42mm and a wireless transaction-authentication device [12]. The BlueBorne family of 8 zero-day vulnerabilities was previously reported to impact Android, iOS, Windows and Linux Bluetooth implementations, despite over 8.2 billion devices being in use [10] and wider surveys calling out attacks on LE devices in Internet-of-Thing and wearables scenarios [11]. The security outcomes are thus link dependent: the same handset could pair to a secure handset with one accessory, and to another, less secure handset, with another accessory, so a security outcome framework that only records if the pairing was successful would not distinguish these cases.

2.3. Service Discovery, Profiles and Codecs

Once the link is made the behaviour that the user can access depends on service discovery and profile conformance. On BR/EDR, the initiator requests service records from the peer; on LE, the client discovers services and characteristics from the peer's GATT server, the services are a set of characteristics, the characteristics are a single value and may contain zero or more descriptors, and each attribute is identified by a universally unique identifier; the profile defines the behavior of a device in a specific application, and a device can implement more than one [17]. Interoperability in audio is demonstrated by numerous negotiations needed to achieve an audio user visible result: A2DP defines the streaming of audio over a source-sink interface and includes provisions for codec interoperability; AVRCP defines the remote control of media and playback state; HFP defines the control of telephony and audio between an audio gateway and a hands-free unit, with version determining which speech quality modes are supported [4] [23]. The mandatory A2DP codec provides a common baseline, not necessarily the preferred codec. Testing should record the negotiated codec and assess it against a predefined preference policy.

2.4. Platform and Stack Heterogeneity

The platform layer exposes the specification behaviour to applications, and the differences in platform policy are what can affect the test outcomes. Android exposes the stack via classes in the android.bluetooth package: BluetoothAdapter represents the local radio, and is the starting point for discovery and to query bonded devices; BluetoothDevice represents a remote device and exposes its name, address, class and bonding state; BluetoothSocket and BluetoothServerSocket are connection endpoints; and profile proxies such as BluetoothA2dp and BluetoothHeadset, which supports both Headset and Hands-Free profiles, represent individual profiles [16]. According to Android pairing is discovery, pairing request, key exchange and caching, data exchange, release the channel as long as the bond remains, reconnection happens automatically when the devices are in range and a bond hasn't been removed [16]. It records that data transferred via a paired LE link is available to all applications on the handset; therefore, applications that process sensitive data should employ application-layer security; it also differentiates the central from peripheral roles at the connection level and the GATT client from GATT server roles at the data level, stating that two peripherals cannot communicate with each other, nor can two centrals [17]. Apple's Core Bluetooth is described as an abstraction of the LE protocol stack based on the Bluetooth 4.0 specification, in which a central scans for peripherals that advertise data and a peripheral responds to the central; since macOS 10.9 and iOS 6, Apple devices can themselves act as LE peripherals [18]. According to Apple documentation, a background or suspended application cannot, by default, perform one of the LE tasks unless it declares a Core Bluetooth background execution mode; certain tasks behave differently in the background even when it declares such a mode; and state preservation and restoration of central and peripheral managers is available since iOS 7 [18], [19]. These are not defects, but represent policy and API limits, and are what a harness can see and control, and why a same accessory can have different behavior from one handset family to another.

2.5. Testing Regimes: Conformance, Profile Testing and Interoperability

Bluetooth qualification defines the compliance requirements and the process by which a product enters the qualified products database [3]. The Qualification Program Reference Document differentiates conformance test cases from Interoperability Testing, which are test cases that are based on the implementation conformance statement for the design and are intended to verify interoperability for core layers [3]. Test cases are assigned to categories: categories A and B are

compulsory for the corresponding layers, such as the radio frequency, baseband, link layer, link manager, and lower Host Controller Interface roles, while categories C, D and X are optional, with category X being called Supplementary interoperability tests, which apply where there is no specification reference [3]. Verdicts are reported as pass or inconclusive (unable to obtain a pass verdict), waived or not required; if a pass verdict cannot be achieved, the member must update the product and retest it successfully, or provide a test coverage waiver, and errors from the test plan, test case and test system are reported as test set erratum [3]. This vocabulary is important as it sets the precedent that non-conformance can be formally waived and that the waiver constitutes evidence, not the lack of evidence. The Bluetooth SIG also provides interoperability test tools for product-to-product testing beyond the conformance framework [20] and test-bed infrastructure for conformance and interoperability testing beyond Bluetooth [22]. In reality the testing of profiles is executed using a Bluetooth Profile Tuning Suite application for the Windows platform that is normally run manually in a GUI [15]. The auto-pts framework automates it using its control COM API, and has been used to automate over 630 test cases across the BlueZ, Zephyr BLE and Mynewt NimBLE stacks, meaning that an effort taken by one person for two months has been accomplished in 15 hours and has been able to support scenarios like Bluetooth Mesh, where two instances of the test system are required to run concurrently [15]. This is the best quantitative automation result found here, that is, the bottleneck in the profile testing is not test specification, but rather execution and bookkeeping, even as the result does not cover cross-vendor product-to-product interoperability. It is common to blur the lines between the regimes when reporting. Conformance testing is simply asking is this conforming to the specification, on a test system; profile testing asks is this profile's required and optional behaviour being implemented and exercisable; interoperability testing asks is this two independently implemented products working together; end-to-end functional compatibility asks is this end-to-end functional use case completed; user perceived reliability asks is this end-to-end functional use case completed over time, across reconnections and under adverse radio conditions. A product can pass the first two and fail the last three.

2.6. Related Work

Table I contrasts the major sources examined by focus, methodology, devices or profiles examined, findings, stated limitations and relevance. The best evidence is either security-based, covering particular attacks targeting individual protocol phases, or performance based, describing timing behaviour under controlled situations, but not necessarily the completion of functionality; no reviewed work offers a framework for a coverage model that spans the entire connection life cycle, with documented denominators and observation windows that the framework in Section IV is suggested to provide. Work on link behaviour sets the stage for the importance of controlling environment. A model of LE discovery derived from advertising and scanning parameters, validated by simulation over ten runs per scenario with advertiser and scanner counts from one to ten, reports that mean discovery latency increases linearly with the advertising interval, decreases gradually with the scan window while it is below 100 ms and then remains almost constant, is largely unaffected by the advertising period per channel, and rises sharply as the number of devices increases because of control-packet collisions [13].

Table 1. Comparison of Related Interoperability Research

Ref.	Year	Research Focus	Methodology	Devices Or Profiles Examined	Key Findings	Limitations	Relevance
[2]	2020	Core Specification 5.2 feature definition: EATT, LE Power Control, Isochronous Channels	Specification feature-overview document	Core Specification 5.2; no devices; no profiles tested	EATT supports concurrent transactions, L2CAP interleaving and dynamic ATT MTU and is restricted to encrypted links; LE Power Control permits receiver-driven transmit-power adjustment; Isochronous Channels target time-synchronised data delivery	Feature description only; no interoperability testing and no measurements reported	Defines the optional-feature negotiation surface that separates an expected configuration from a defect
[5]	2019	Key negotiation weakness in BR/EDR and	Adversarial protocol analysis with	17 Bluetooth chips incl. Broadcom,	Key entropy can be forced as low as 1 byte; all 17	Attacker must be in range and able to intercept	Establishes that link security

		LE encryption setup	implementation testing	Apple, Intel and CSR	chips tested were vulnerable	and manipulate Bluetooth traffic; chip sample is not exhaustive	strength is a negotiated, link-dependent quantity requiring its own test category
[7]	2020	Impersonation during secure connection establishment	Adversarial protocol analysis with device testing	31 devices comprising 28 unique Bluetooth chips	30 of 31 devices vulnerable; both the legacy authentication procedure and the secure authentication procedure are affected	Attacker requires Bluetooth firmware access and a baseband transceiver, and is assumed able to eavesdrop, manipulate and jam	Motivates explicit validation of the pairing method and association model actually negotiated
[8], [9]	2020	Robustness of LE system-on-chip implementations to malformed input	Automated over-the-air fuzzing framework	12 devices from 8 vendors and 4 IoT products	11 new vulnerabilities and 13 CVE identifiers; observed effects include deadlocks, crashes and buffer overflows	Results are conditioned on the specific firmware versions examined; device-level, not pair-level	Motivates malformed-input, timeout and error-handling tests as a first-class category
[12]	2020	Method confusion in the Bluetooth pairing procedure	Protocol analysis with implementation evaluation	Bluetooth pairing procedure; implementation set not stated in retrieved material	Confusion between pairing methods permits a downgrade of the pairing method used	Post-2020 context source; evaluation set and device coverage not established from the retrieved material	Supports verification that the negotiated pairing method equals the intended method
[13]	2015	Bluetooth LE device discovery latency	Analytical model validated by simulation, 10 runs per scenario	Simulated advertisers and scanners, 1 to 10 of each; no hardware	Mean discovery latency increases linearly with advertising interval, falls gradually with scan window until about 100 ms then stays near constant, is little affected by advertising period per channel, and rises sharply as device count grows due to control-packet collisions	Simulation only; no handset or accessory product involved; single-channel timing constants assumed	Defines the parameter sensitivity that requires discovery parameters to be recorded with every run
[14]	2019	Timeliness of connection-based LE communication under interference	Testbed experiment with controlled interference, 10 repetitions per setting	9 single-board-computer nodes and LE development kits in a 6 m × 10 m	For a 250 ms connection interval the expected upper latency bound is 260 ms; up to	Single laboratory; one hardware and stack combination; UDP payload;	Defines the environmental-control requirement and the connection-

				laboratory	21.74% of datagrams exceeded it during a busy daytime period with some exceeding 1000 ms; Bluetooth interference produced at least 2× the bound and Wi-Fi interference near the peripheral at least 5×	results not transferable without re-measurement	stability metric
[15]	2019–	Automation of Bluetooth profile testing	Automation of the Profile Tuning Suite through its control COM API	BlueZ, Zephyr BLE and Mynewt NimBLE stacks; more than 630 test cases	Testing effort reduced from an effort described as one person for two months to 15 hours; Bluetooth Mesh scenarios supported with two concurrent test-system instances	Addresses stack-level profile conformance, not cross-vendor product-to-product interoperability	Establishes feasibility of scripted execution and of automated evidence bookkeeping
[16]	2020	Platform API surface and bonding policy for BR/EDR	Vendor developer documentation	Android Bluetooth API: adapter, device, socket and profile proxies	Pairing is followed by bonding, in which security keys are exchanged and cached so that reconnection occurs automatically while the devices are in range and the bond is intact	Policy and API documentation; no test results reported	Defines the platform metadata fields and the observable control surface of the harness
[17]	2020	Platform API surface, GATT roles and LE privacy behaviour	Vendor developer documentation	Android BLE APIs; GATT, ATT, services, characteristics and descriptors	Central/peripheral roles are distinct from GATT client/server roles, and two peripherals cannot communicate with each other nor can two centrals; data exchanged over a paired LE link is accessible to all applications on the device	Documentation; no interoperability results; privacy behaviour is platform policy	Motivates role-coverage requirements and the privacy validation sub-category
[18], [19]	2019	Platform abstraction and background-execution constraints	Vendor developer documentation and conference session	Core Bluetooth on iOS and macOS; Bluetooth 4.0 basis	Core Bluetooth abstracts the LE stack; a background or suspended application	Documentation; no interoperability results; platform-version	Requires background and foreground application states to be

					cannot by default perform LE tasks unless it declares a background execution mode, certain tasks behave differently in the background even then, and state preservation and restoration has been available since iOS 7	dependent	recorded as test conditions

3. Research Problem and Requirements Analysis

3.1. Definition of the Interoperability Problem

Let H be a handset in a set of handset models, and let A be an accessory in a set of accessory models, defined by a controller, a host stack, an operating system or firmware image, a declared feature set, supported profile versions and a security configuration. The interoperability problem is whether a given handset–accessory pair ((H,A)) can complete a defined use case (U) within specified functional, timing, reconnection, environmental, and security criteria. This is the source of the ambiguity in reporting described in Section II: conformance is a property of a single implementation, whereas interoperability is a property of a combination and the number of combinations grows multiplicatively with the number of models in each population.

3.2. System Boundaries and Objectives

The system to be tested is the handset-accessory connection and the profile sessions that are transferred through it. The boundary covers: Controller, host stack, operating-system Bluetooth service, application or system component, and the radio channel between the application or system component and the operating-system Bluetooth service. It excludes cloud services behind the handset application, the acoustics and amplifier performance of the accessory, and regulatory radio compliance, each separately governed. The objectives are to make interoperability outcomes reproducible, coverage explicit, failure causes attributable, and regression detection routine rather than episodic.

3.3. Testing Dimensions

The outcome space is determined by six dimensions and needs to be recorded for each run. The handset dimension comprises model, operating-system family and version, host stack identity, controller vendor and firmware revision, and transport selection. The accessory dimensions include model, firmware revision, declared profiles, codec set and supported roles. The version dimension contains the Core Specification versions that are supported by each side and the profile versions that are actually negotiated, not the profile versions that are declared. The security dimension includes supported pairing methods, association models and minimum key strength. The configuration dimension comprises bonding state, number of concurrent connections and platform policy such as background-execution restrictions. The environmental dimension comprises distance, orientation, ambient 2.4 GHz activity and physical obstruction. The environmental dimension is not something that can be ignored if the results are to be comparable over runs since it creates mean discovery latency that scales rapidly with the number of contending devices [13] and co-located Wi-Fi interference that produces a discovery latency of at least five times the bound [14].

3.4. Distinguishing Defects from Unsupported Features

The framework classifies all non-success observations into four categories since these four classes are the most common culprits for over or under represented success rates. An interoperability defect is a non-success in a scenario whose expected outcome is required by a specification to which both devices declare conformance. A degraded result is a failure in which the required fallback function was used, such as reducing the required audio codec to an optional higher quality codec supported by both devices. An unsupported feature is a scenario where a pre-condition is not met because one device has not declared the requirement, it is not counted in the denominator and is marked as unsupported. An indeterminate outcome is an outcome for which a harness fault was identified before the run was complete, or an environment excursion or unrecoverable platform state was identified before the run was complete, and is not included in either the numerator or denominator. Interoperability defects and degraded outcomes are included in the denominators of applicable success-rate metrics, but only successful outcomes contribute to their numerators. Unsupported scenarios are reported separately and excluded from the denominator when the required preconditions are not supported. Indeterminate runs are excluded from both numerator and denominator and reported separately. These rules ensure that every reported metric has a reproducible denominator.

3.5. Requirements

Table II lists fourteen test categories, numbered INT-01 to INT-14, with preconditions, procedure, expected outcome, evaluation metric, evidence source and pass/fail criteria, and indicates for each whether it is derived from the specification or is a proposed engineering criterion. The vocabulary of pass, inconclusive, waived and not required [3] used for Bluetooth remains consistent throughout the framework, while degraded, unsupported and indeterminate are added for outcomes that are not included in the Bluetooth qualification process.

Table 2. Interoperability Requirements and Test Scenarios (Proposed)

Id	Test Category	Preconditions	Test Procedure	Expected Outcome	Metric	Evidence Source	Pass/Fail Criteria
INT-01	Discovery and connection initiation	Both devices powered; accessory in each declared discoverable state; environment profile verified	Instruct each side to discover the other in turn; record advertisement or inquiry parameters and time to first sighting; repeat n times	Peer is discoverable in every declared state and is observed within the scenario timeout	(2) L_est; discovery success count	Spec.-derived (core inquiry and advertising behaviour); [13] for parameter sensitivity	PASS if all declared states are discovered within timeout; DEGRADED if only some states are discovered; FAIL if none
INT-02	Pairing	No existing bond; pairing enabled on both sides; association-model capability recorded for each side	Initiate pairing for each association model the pair jointly supports; record the model negotiated and the outcome	Pairing completes and the negotiated association model equals the strongest model the pair jointly supports	(1) PSR	Spec.-derived (pairing and association models) [1]; [5], [12] for manipulation risk	PASS if PSR = 100% and no silent downgrade; DEGRADED if a permitted weaker model is negotiated; FAIL on error or timeout
INT-03	Bonding and key persistence	Pairing (INT-02) has succeeded; bond created	Power-cycle both devices; query the bonded-device list on the handset; reconnect without re-pairing	Bond persists across the power cycle and the peer appears in the bonded-device list with the same identity	Bond-persistence rate (count-based)	Spec.-derived (bonding and key storage) [1]; [16] for platform behaviour	PASS if the bond persists and the peer is listed; FAIL if the bond is lost or the peer identity changes
INT-04	Authentication and association model	A bond exists; both sides expose their pairing capabilities	Establish a secure connection using the existing bond; record the authentication procedure used and the resulting key strength	Authentication completes using the strongest procedure both sides support and the key strength meets the configured minimum	Secure-establishment rate (count-based)	Spec.-derived (legacy vs. secure authentication) [1]; [7] for impersonation risk	PASS if the strongest procedure is used and the minimum key strength is met; DEGRADED if a permitted weaker procedure is used; FAIL otherwise
INT-05	Service discovery	Link established; peer service or profile declaration	Enumerate the peer's service records (BR/EDR) or services,	Every declared service, characteristic or record is	(3) SDR	Spec.-derived (service discovery and GATT) [17]; [15] for	PASS if all declared items are retrieved; DEGRADED if a declared

		recorded	characteristics and descriptors (LE); compare with the declared set	retrieved within the scenario timeout		automation	optional item is missing; FAIL if required items are missing or the discovery times out
INT-06	Profile compatibility	Both devices declare their supported profile versions and optional features	Construct the pair profile matrix; exercise each profile in the negotiated intersection and record the optional features actually used	Each profile in the intersection is usable and the optional features in use are those both sides support	Profile-compatibility rate (per profile)	Spec.-derived (profile versions) [4]; [23] for profile procedure	PASS if every profile in the intersection is usable; DEGRADED if a permitted optional feature is not used; FAIL if a profile in the intersection is unusable
INT-07	Audio streaming (A2DP)	A2DP declared on both sides; audio route selected; source material and payload set defined	Stream the defined payload set for the defined duration; record the negotiated codec and measure interruptions and audio-gap events	Streaming completes for the full duration using the best codec the pair jointly supports	(5) DTCR applied to audio frames; audio-gap count per hour	Spec.-derived (A2DP roles and codec interoperability) [4]	PASS if streaming completes with the best common codec; DEGRADED if a permitted fallback codec is used; FAIL on interruption or stream loss
INT-08	Audio control and telephony (AVRCP, HFP, HSP)	Declared control and telephony profiles present; call-control and media-control operations enumerated	Exercise each declared control operation (play, pause, track change, volume, call answer, call end) and record the result of each	Every declared control operation produces the expected state change on the peer	Control-operation success rate (count-based)	Spec.-derived (AVRCP, HFP, HSP) [4]; [23] for HFP procedures	PASS if all declared operations succeed; DEGRADED if only optional operations fail; FAIL if a required operation fails
INT-09	Data transfer	Declared data transport available; payload set, integrity check and observation window defined	Transfer the defined payload set over the declared transport; verify delivery, ordering and integrity	Every payload is delivered intact within the observation window	(5) DTCR	Spec.-derived (GATT and serial transport) [17]	PASS if all payloads are delivered intact within the window; FAIL on loss, corruption or timeout
INT-10	Reconnection and automatic re-establishment	Bond intact for the first condition and removed for the second; range-control	Move the accessory out of range and back; separately power-cycle	The link re-establishes automatically while the bond is intact, and re-pairing	(4) RSR; (7) T_rec for the reconnection mode	Spec.-derived (bonding behaviour) [1]; [16] for platform reconnection	PASS if re-establishment succeeds within timeout for the intact-

		mechanism available	it; record time to re-establishment and the number of attempts	succeeds after bond removal		policy	bond condition; FAIL if manual intervention is required
INT-11	Multi-connection and coexistence	Two or more accessories available; handset supports the required concurrent links	Establish concurrent links to the defined accessory set; exercise each use case while the others are active	Each use case completes while the other links are active, and no link is lost as a result of the others	(5) DTCR per link; (6) CDF	Proposed engineering criterion; platform role constraints from [17]	PASS if all use cases complete with no link loss; DEGRADED if a permitted link priority change occurs; FAIL on link loss or use-case failure
INT-12	Connection stability and environmental robustness	Environment profile verified and recorded; defined load profile and observation window	Hold the link under the defined load for the observation window; repeat under each declared environmental condition and report per condition	No unintended disconnection occurs during the observation window and timeliness remains within the scenario bound	(6) CDF; (2) L_est per condition	Proposed engineering criterion; sensitivity evidence from [13], [14]	PASS if CDF = 0 and latency remains within the bound; DEGRADED if latency exceeds the bound without disconnection; FAIL on unintended disconnection
INT-13	Error handling, timeout and recovery	Failure-injection mechanisms available for each declared failure mode; scenario timeout defined from expected completion time	Inject each declared failure mode (out of range, peer power loss, harness disconnect, malformed input) and measure recovery	The pair returns to the pre-failure functional state within the scenario timeout and no persistent fault remains	(7) T_rec per failure mode	Spec.-derived for input validation; [8], [9] for robustness evidence	PASS if recovery occurs within timeout for every mode; FAIL if a mode requires manual intervention or leaves a persistent fault
INT-14	Security and privacy validation	Pairing capability recorded; configured minimum key strength defined; application data path identified	Record the negotiated pairing method and key strength for each pair; verify encryption is active before application data; verify reachability of paired-link data from other applications;	The strongest jointly supported method is used, the minimum key strength is met, encryption precedes application data, and sensitive payloads are not exposed to unrelated	Secure-negotiation rate (count-based); key-strength conformance	Spec.-derived (pairing, authentication and key strength) [1]; [5], [7], [11], [12] for risk; [17] for privacy behaviour	PASS if all four checks hold; DEGRADED if a permitted weaker method is negotiated; FAIL if the minimum key strength is not met, if encryption is absent, or if sensitive data is exposed

			test re-pairing and post-bond-removal behaviour	applications			

4. Proposed Bluetooth Interoperability Test Framework

The framework consists of 13 elements arranged in a pipeline from inventory to execution to reporting, and a loop of feedback from failure analysis to inventory back to test selection. Components 1–3 are for setting up and validating the test configuration, components 4–9 are for running the fourteen test categories of Table II, which include the objective, preconditions, procedure, expected outcome, failure conditions, required logs and evaluation metric for each category; components 10–13 convert raw execution into classified, traceable, and reproducible evidence. The architecture is presented in Fig. 2.

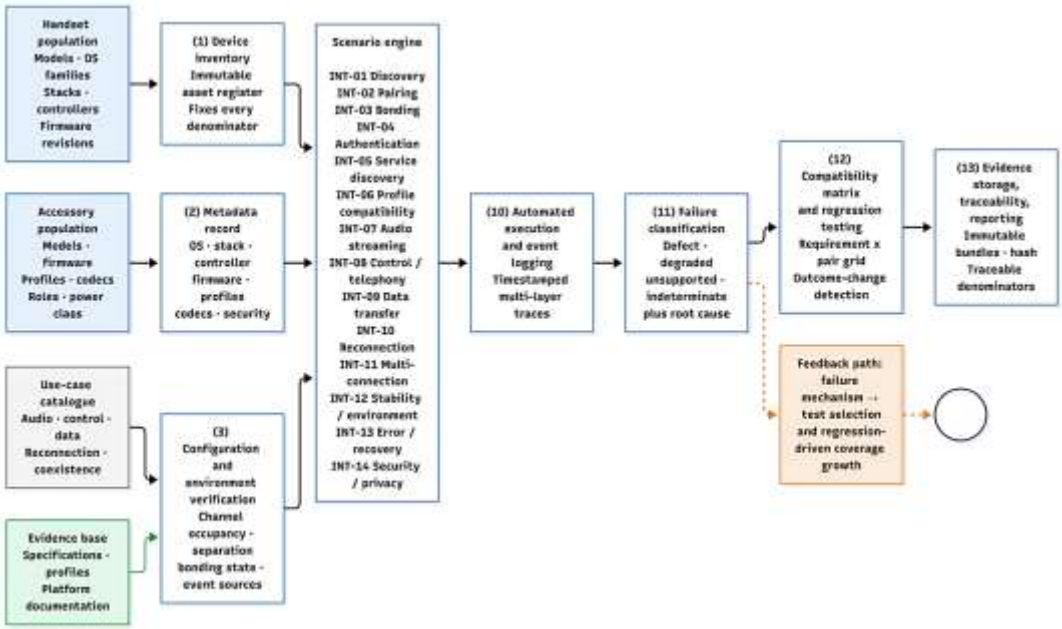


Figure 2. Proposed Bluetooth Interoperability Test Framework, From Inventory and Metadata Through Orchestration, Scenario Execution, Classification and Reporting

4.1. Inventory and Metadata (Components 1 and 2)

Component 1 is an ongoing database of all devices accepted to the programme, identified by an unchanging asset identifier and populated with details such as model, unit identifier, owner, acquisition date and lifecycle state; each entry contains the metadata record and connects to each run in which the device was used, and the inventory defines the population from which coverage is planned; the denominator for each metric is defined separately. Component 2 is a machine-readable metadata record which includes the six dimensions of Section III-C: handset, model, operating-system family and version, host stack identity, controller vendor and firmware revision, transport support and declared profiles; accessory, model, firmware revision, declared profiles and versions, codec set, roles and power class; and pair, bonding history, any recorded waiver. The behaviour of the platform is policy-dependent, and the record also includes information about the interaction between the harness and the stack as the same handset might expose different control surfaces to a system component and to a third party application [16]–[18].

4.2. Configuration Verification and Scenario Execution (Components 3 to 9)

Component 3 is used to verify the environment before the execution of a scenario: Radio channel occupancy in the 2.4 GHz band is sampled and recorded, separation and orientation of the pair is fixed, bonding state of both devices is set to the value the scenario requires and the harness verifies that it can read the event sources on which the metrics depend. The profile is stored with the run so that a later run can be declared comparable or not, because the reported sensitivity of LE timeliness to interference [14] means an unverified environment yields results that cannot be attributed to the devices.

Component 4 executes the discovery, pairing, bonding, authentication and reconnection categories (INT-01 to INT-04 and INT-10). Discovery is exercised in both directions with both advertisement and inquiry parameters recorded, for each declared discoverable state, thus allowing for a measure of discovery latency based on the parameter sensitivity reported in [13]. Pairing is performed for each association model that the pair can support together, and the model which is actually negotiated is saved

and contrasted with the highest model the pair could support, downplaying being documented as degraded or as a defect depending on whether the specification requires the stronger model. Bonding is verified by ensuring that the key is still present after a power cycle and reconnection is exercised while the bond is active and once the bond is removed. Fig. 3 shows the sequence and its decision points.

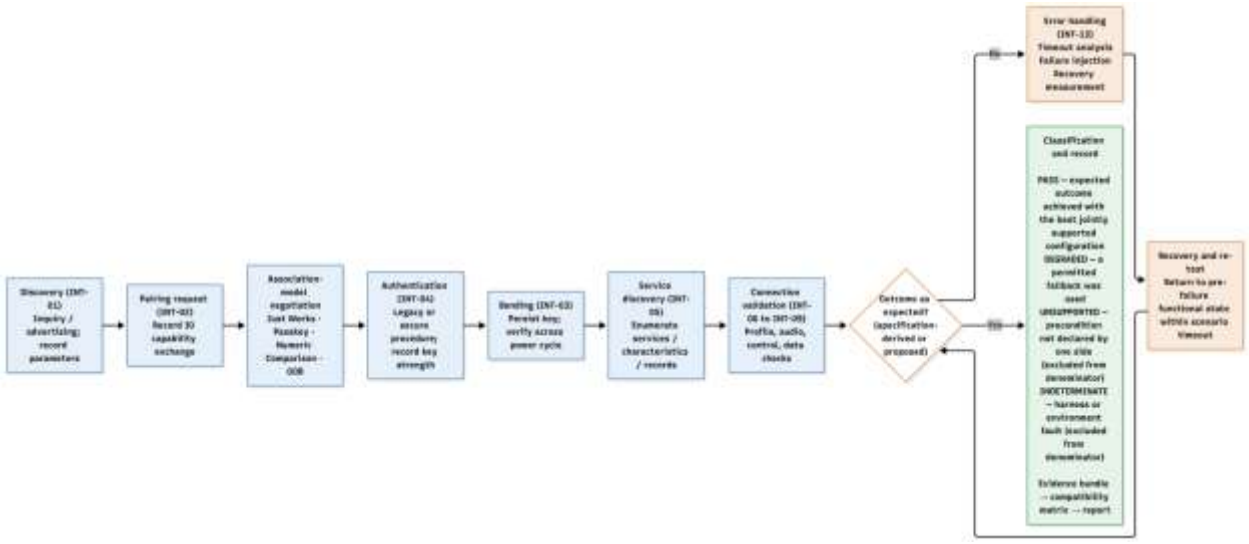


Figure 3. Bluetooth Pairing and Interoperability Test Workflow, Including Error Handling, Recovery and Pass/Fail Classification

The service discovery and profile compatibility categories (INT-05, INT-06) are the categories executed by Component 5, which enumerates the peer's services and characteristics, and compares the result to the peer's declared profile and service set performed at the level of services, characteristics and descriptors [17] and tests the negotiated intersection of profile versions and optional features, rather than the declared union; since profile versions in the field cover several generations of the same profile [4], it is here that the highest number of legitimate but degraded outcomes is expected. Component 6 implements the audio, control and data transfer categories (INT-07 to INT-09): audio streaming is validated with respect to the codec actually negotiated; control is validated with respect to the media and telephony operations the pair agrees to; and data transfer is validated with respect to the declared transport, which is then checked with completion, throughput and integrity over a defined payload set and observation window, and with a degradation failure counted as a failure.

Component 7 implements the categories connection stability and environment (INT-11 and INT-12) by keeping the link stable under a representative load for a specified time and recording the events of the connection, by comparing the number of unintended disconnections with the time the connection is stable, and by repeating a set of defined scenarios under controlled radio conditions, with radio conditions reported per condition instead of pooled, because the number of radio conditions with different interference levels would produce a quantity without any physical interpretation. The error handling timeout and recovery (INT-13) category (Component 8) defines a timeout value that is determined by the task to be performed; the timeouts are considered as events that are separate from the protocol errors and disconnections; recovery is evaluated as the time required to return to a functional state after failure, including the handling of malformed and unexpected inputs, because over-the-air fuzzing of LE system-on-chip implementations has resulted in deadlocks, crashes and memory corruption in a significant percentage of the products tested [8], [9].

Security and privacy is a category (INT-14) and Component 9 performs the following security tests: Security Outcomes records the pairing method and association model negotiated for each pairing; Security Outcomes is separated from functional testing because security outcomes are link-dependent; Security Outcomes records the pairing method and association model negotiated for each pairing; Security Outcomes verifies that the negotiated model matches the strongest one the pair can achieve; Security Outcomes checks that key strength meets the minimum key strength configured; Security Outcomes confirms that the link is encrypted before application data is exchanged. It also checks behaviour on re-pairing and after bond removal: manipulation of the authentication phase has been shown to allow impersonation of a bonded peer without possession of the long-term key [7]; manipulating the pairing method has been shown to allow a man-in-the-middle position without compromising the cryptography of either pairing method [12]; and manipulation of key negotiation has been shown to imply a reduction of key entropy to as low as 1 byte [5], [6]; on privacy, it checks whether data exchanged over a paired LE link is available for other applications on the handset, which platform documentation says is the default behaviour, and which motivates the application-layer protection for sensitive payloads [17].

4.3. Failure Classification and Compatibility Analysis (Components 10 to 12)

Component 10 automates execution wherever the platform exposes a control surface and records for every run the scenario identifier, parameter set, environment profile, device metadata versions and a timestamped event stream from each observable layer; automation of profile-level testing has been shown to reduce an effort described as one person for two months to fifteen hours for more than 630 test cases across three stacks [15], and the framework extends the same principle to product-to-product runs, where the observable event sources are platform logs, host-controller traces and radio captures. In Component 11, each occurrence of a failure is classified against the recorded event stream, instead of the tester's memory, and each failure is classified into one of the four classes of Section III-D and, if it is a failure or a reduced outcome, to a root cause category: negotiation, timing, capability mismatch, state management, input validation or environment, and the classification feeds the feedback path of Fig. 2, which chooses which scenarios should be added or strengthened if they are implied by a failure. The compatibility matrix is maintained by component 12; it stores the evidence identifier, negotiated configuration and outcome class for each pair and test category, and is updated on re-run; thus, any change in outcome class for a previously passing pair is a regression. Regression scope is where the changed component and the pairs that depend on that component meet up, so that a firmware change on one accessory only affects those pairs that include that accessory and not the entire inventory.

4.4. Evidence Storage, Traceability and Reporting (Component 13)

Each run generates an immutable evidence bundle of event streams, environment profile, metadata snapshots, classification, and the resulting matrix entry, addressed by a content hash with a link to exercised requirement identifiers. Reports are made from the matrix and bundles and not prepared manually, which means that for every reported number there is a traceable denominator. This component also maintains the qualification vocabulary of verdicts that are run away and verdicts that are not required [3] as they are evidence and should not be silently changed to an untested scenario.

5. Research Methodology and Experimental Design

5.1. Research Design and Literature Strategy

The present study is a framework-design and a synthesis of evidence study rather than an experimental campaign. The literature search included the Bluetooth Core Specification, the adopted profile and service specification register, the literature of the qualification programme, platform vendor developer documentation for the Android and Apple platforms, and peer-reviewed and pre-print literature on Bluetooth security, link performance, discovery modelling and test automation. Sources were cited only if they were retrieved in full text, and if the bibliographic identity could be established in the retrieved document; sources whose full text could not be retrieved were not cited, but the failures to retrieve are revealed in Section VIII. The technology baseline is the 2020 period, and sources published after 2020 are used only where explicitly labelled as later context.

5.2. Test-Case Construction and Device Selection

Test cases were developed by breaking down the connection life cycle into the fourteen categories outlined in Table II, and in each, enumerating the degrees of freedom of Section III-C. A prospective campaign will be stratified across the handset and accessory populations, with a minimum of three handset families and four accessory classes with at least one device per side and per profile version, and per Core Specification generation. Because no campaign was executed, this is stated as a requirement on the methodology rather than a description of performed work.

5.3. Metric Definitions

Eight formal metrics are defined, with explicit numerators, denominators, units, observation windows, and interpretation rules. For success-rate metrics, valid failed and degraded attempts remain in the denominator, while only successful outcomes enter the numerator. Unsupported scenarios and indeterminate runs are reported separately and excluded where appropriate. Pairing success rate is the proportion of valid pairing attempts that establish a bond:

$$PSR = (N_{pair,succ} / N_{pair,valid}) \times 100\% \qquad (1)$$

counted over the run set for a single pair and association model, where the denominator excludes unsupported and indeterminate attempts. Connection establishment latency is reported as the mean with the 95th percentile, since the distribution is right-skewed in the presence of retries:

$$L_{est} = mean(L_{i}) , \quad L_{95} = Q_{0.95}(L_{i}) \quad [ms] \qquad (2)$$

measured from the first transmission of the connection request to the first successful application-level response. Service discovery success rate is

$$SDR = (N_{sd,succ} / N_{sd,valid}) \times 100\% \qquad (3)$$

Where success requires retrieval of every service, characteristic or service record the peer declares, within the scenario timeout. Reconnection success rate is

$$RSR = (N_{recon,succ} / N_{recon,valid}) \times 100\% \quad (4)$$

With the window opening when the accessory returns to range and closing at the scenario timeout, reported separately for intact-bond and bond-removed conditions. Data transfer completion rate is

$$DTCR = (N_{dt,succ} / N_{dt,valid}) \times 100\% \quad (5)$$

Where success requires delivery and integrity verification of every payload in the defined set. Connection drop frequency is a rate, reported in events per hour:

$$CDF = N_{drop} / T_{obs} \quad [h^{-1}] \quad (6)$$

With T_obs the accumulated connected time and with harness-caused drops excluded. Recovery time is the mean interval between the injected failure and the return to the pre-failure functional state:

$$T_{rec} = (1/m) \Sigma_{j=1}^{m} T_{rec,j} \quad [s] \quad (7)$$

Reported per failure mode, because recovery behaviour is not exchangeable across modes. Test coverage is defined on the requirement-by-pair grid:

$$Cov = (N_{executed} / N_{planned}) \times 100\% \quad (8)$$

Where N_planned is the number of requirement-and-pair combinations selected by the sampling strategy and N_executed the number whose outcome class is success, degraded or unsupported; indeterminate runs are excluded from N_executed and counted separately, so a harness fault cannot be presented as coverage. Finally, the expected outcome of a scenario is a deterministic predicate over the pair's negotiated configuration rather than a probabilistic claim:

$$Outcome(H, A, U) = f(Feature(H) \cap Feature(A), U) \quad (9)$$

Where Feature_neg(H, A) denotes the set of capabilities negotiated by handset H and accessory A on the tested link, and U denotes the use case. Equation (9) formalises the principle that the expected outcome depends on the negotiated capabilities of the pair and the requirements of the use case, rather than on either device's declared capabilities alone.

5.4. Handling of Missing Data, Unsupported Features, Timeouts and Invalid Runs

A run that fails environment verification, for which no harness is defined or when the harness does not know what to do in another platform state, will be indeterminate. A run is unsupported when a pair has declared a precondition that is not supported by the pair; such runs are not counted in denominators and are reported as a separate count naming the precondition that is not supported. A timeout is an event independent of protocol error and considered a failure only in situations for which the expected result is specified-derived. Every metric is presented with the number of runs excluded, meaning that the denominator is always reconstructable and where the measurement cannot be obtained on a given platform it is reported as not measurable for that pair, not imputed.

5.5. Coverage and Evidence Analysis

Coverage is assessed on the two-dimensional grid of requirement identifier against pair, with each cell assigned to one of four states: planned, executed, unsupported or untested. The assessment structure is depicted in Fig. 4, with the evidence traceability path indicated for each cell and the evidence that supports the cell indicated in the assessment bundle. The figure refers to categorical states only; there is no percentage coverage shown, as there was no campaign and no percentage would be an invention.

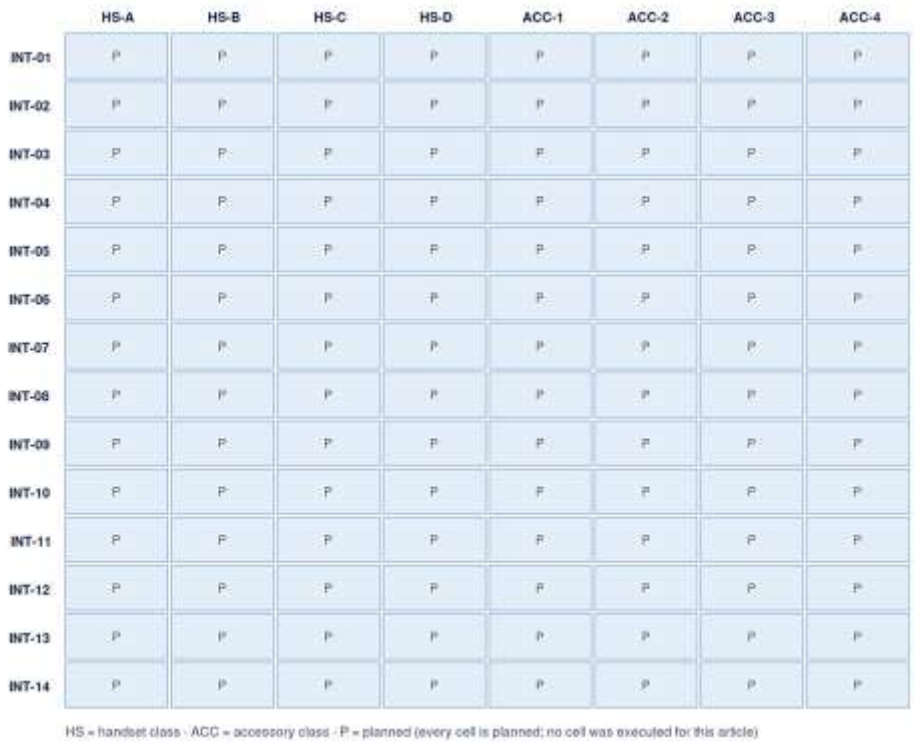


Figure 4. Interoperability Test Coverage and Evidence Analysis, With Cells Marked Planned, Executed, Unsupported or Untested

6. Results

6.1. Literature-Derived Interoperability Challenges

Four regular sorts of challenge arise. The first is capability negotiation, depending on the combination of two independently-stated feature sets, not either on the first feature set or the second, as formalised in (9) and illustrated by the presence in the field of multiple generations of the same profile [4] as well as negotiation of optional Core Specification features like EATT and LE Power Control [1, 2]. The second is security negotiation, in which link strength is agreed in-band and has been shown to be manipulable, both in key entropy, where all seventeen chips tested were vulnerable [5], [6], and in authentication and pairing method, where thirty of thirty-one devices and a set of off-the-shelf handsets, a smartwatch and a payment device were respectively vulnerable [7], [12]. The third one is implementation robustness: malformed over-the-air input causes deadlocks, crashes and/or memory corruption in a significant fraction of the LE system-on-chip implementations tested, resulting in eleven new vulnerabilities, and thirteen CVE identifiers in twelve devices from eight vendors [8], [9]. The fourth is environmental sensitivity (an example is given below), where the same pair shows a materially different timing behaviour depending on ambient 2.4 GHz activity, with measured latencies being up to five times the bounds, and discovery latency increasing rapidly with the number of contending devices near the peripheral [14,13].

6.2. Cross-Device and Cross-Profile Compatibility Requirements

Table III combines the quantitative results that are reported by the sources retrieved, includes the page or section, the device or test context, the reported value, the conditions, the interpretation used here and the limitations on this interpretation. Three requirements follow. The negotiated configuration should be documented for each run as the same pair may have different results with the same negotiation. The two pairing methods and the model of association used must be noted and contrasted with the best model the two can support, since a successful pairing does not prove that they have attained a secure pairing. The ambient radio condition should be noted and results should not be combined across radio conditions; the reported latency distributions vary by a factor of five between radio conditions in a single study [14].

Table 3. Source-Reported Findings and Quantitative Evidence

Ref.	Document Page or Section	Device Or Test Context	Reported Metric	Reported Value	Test Conditions	Interpretation	Evidence Limitations
[2]	§ 1–§ 3 (feature overview)	Core Specification 5.2 feature set	Feature capability (qualitative)	Not reported (qualitative)	Specification feature description; no test performed	EATT permits concurrent transactions, L2CAP	Descriptive document; no measurement and no

						interleaving and dynamic ATT MTU and is restricted to encrypted links	interoperability test
[5], [6]	Results; device-coverage table	17 Bluetooth chips incl. Broadcom, Apple, Intel, CSR	Vulnerable chips	17 of 17 tested (100% of the tested sample)	Attacker in radio range, able to intercept and manipulate Bluetooth traffic	Encryption-key entropy is negotiated in-band and can be forced as low as 1 byte	Convenience sample of chips; proportion is of the tested sample only, not of the installed base
[7]	§ VI; device-coverage table (Table III in source)	31 devices comprising 28 unique Bluetooth chips	Vulnerable devices	30 of 31 tested (≈ 96.8% of the tested sample)	Attacker has Bluetooth firmware access and a baseband transceiver, and can eavesdrop, manipulate and jam	Impersonation of a bonded peer is possible without possession of the long-term key during secure connection establishment	Single device not vulnerable (ThinkPad 41U5005 mouse); convenience sample; results are device-level
[8], [9]	Abstract; § 1 (introduction)	12 devices from 8 vendors and 4 IoT products	New vulnerabilities and CVE identifiers	11 new vulnerabilities; 13 CVE identifiers	Automated over-the-air fuzzing of LE system-on-chip implementations	Malformed or unexpected over-the-air input produces deadlocks, crashes and buffer overflows	Conditioned on the specific firmware versions examined; firmware updates can remove a defect without changing the model designation
[13]	§ 4–§ 5; simulation parameter table	Simulation, 10 runs per scenario; 1–10 advertisers and 1–10 scanners	Mean discovery latency trend	Increases linearly with advertising interval; falls gradually with scan window below about 100 ms then near constant; largely unaffected by advertising period per channel; rises sharply with device count	Advertising interval 30–10,240 ms; scan interval and window 30–10,240 ms	Discovery latency is a parameter- and contention-dependent quantity, not a device constant	Analytical model validated by simulation, not by hardware measurement; no handset or accessory product
[14]	§ V (evaluation)	9-node testbed in a 6 m × 10 m laboratory; LE development kits; 10 repetitions	Datagrams exceeding the latency bound	Up to 21.74% of datagrams exceeded the 260 ms bound during a busy daytime period over 15 minutes;	Connection interval 250 ms; expected upper bound 260 ms; UDP payload; common office environment	Interference degrades connection-based LE timeliness well beyond the model prediction	Single laboratory; one hardware and stack combination; payload and traffic pattern specific to the

		per setting		several exceeded 1000 ms			study
[14]	§ V (controlled interference)	Same testbed under controlled Bluetooth and Wi-Fi interference	Latency under interference	At least 2× the bound under Bluetooth interference; at least 5× the bound under Wi-Fi interference near the peripheral	Interference generated by co-located radios in the same laboratory	Ambient 2.4 GHz activity is a first-order variable that must be recorded and not pooled across conditions	Same limitations; interference levels are laboratory-specific and not calibrated against a standard profile
[15]	Project README (automation scope)	BlueZ, Zephyr BLE and Mynewt NimBLE stacks; more than 630 test cases	Automation effort reduction	Testing time reduced from an effort described as one person for two months to 15 hours	Automation of the Profile Tuning Suite through its control COM API	The dominant cost of profile testing is execution and bookkeeping rather than test specification	Stack-level profile conformance; does not address cross-vendor product-to-product interoperability
[21]	Market update (device shipments)	Bluetooth audio devices	Units shipped and forecast	152 million Bluetooth earbuds shipped in 2020; projected to reach 521 million annual device shipments by 2025	Industry forecast published by the Bluetooth SIG	Establishes the scale of the handset–accessory pairing problem	Forecast figures, not measurements; vendor-reported shipment aggregation
[3]	Qualification Program Reference Document	Qualification process	Test verdict vocabulary	pass, inconclusive, waived, not required	Bluetooth qualification programme requirements	A waiver is recorded evidence of non-testing and must not be converted into an untested scenario	Process document; contains no device measurements

6.3. Failure Categories and Test-Coverage Analysis

When comparing the documented failures to the fourteen test categories in Table II, it can be seen that the evidence base covers a narrow band of the life cycle. The security findings [5,7,8,12] map to INT-02, INT-04, INT-13 and INT-14; the link-performance findings [13,14] map to INT-01 and INT-12 (via the environment-verification dependency), and the automation result [15] maps to INT-05, INT-06, INT-07 and INT-08 at the stack level. No reviewed source reports product-to-product results for INT-10, the reconnection category, despite reconnection dominating a user's daily experience of a bonded accessory, and no reviewed source reports coverage of multi-connection coexistence (INT-11) for a heterogeneous pair. The proposed coverage matrix is shown in Table IV, indicating each cell as proposed but not executed for all fourteen categories and handset and accessory dimensions.

Table 4. Proposed Interoperability Test-Coverage Matrix

Test Category	Handset Dimension	Accessory Dimension	Scenario	Expected Outcome	Required Log	Coverage Status
INT-01 Discovery and connection initiation	OS family and version; controller vendor and	Model; firmware revision; declared	Discover the peer in each declared state, in both directions	Peer observed in every declared state within the scenario timeout	Inquiry or scan trace with parameters; timestamped	Proposed – not executed

	firmware; transport (BR/EDR, LE, dual-mode)	discoverable states			sightings	
INT-02 Pairing	OS family and version; pairing capability; platform pairing policy	Model; firmware revision; IO capability and association-model support	Pair for each jointly supported association model	Negotiated model equals the strongest jointly supported model	Pairing event trace; negotiated method and capability exchange	Proposed – not executed
INT-03 Bonding and key persistence	OS family and version; bonded-device list behaviour	Model; firmware revision; key-storage behaviour	Power-cycle both sides and reconnect without re-pairing	Bond persists and peer identity is stable	Bonded-device listing; reconnect trace	Proposed – not executed
INT-04 Authentication and association model	OS family and version; supported authentication procedures; key-strength policy	Model; firmware revision; supported authentication procedures	Establish a secure connection on an existing bond and record the procedure used	Strongest jointly supported procedure used; minimum key strength met	Authentication trace; negotiated procedure; key strength	Proposed – not executed
INT-05 Service discovery	OS family and version; host stack identity; declared services	Model; firmware revision; declared services and characteristics	Enumerate peer services or records and compare with the declaration	All declared items retrieved within the scenario timeout	Service enumeration dump; declared-versus-discovered diff	Proposed – not executed
INT-06 Profile compatibility	OS family and version; declared profile versions; optional-feature set	Model; firmware revision; declared profile versions and optional features	Exercise each profile in the negotiated intersection	Every profile in the intersection is usable	Per-profile exercise trace; optional-feature usage record	Proposed – not executed
INT-07 Audio streaming (A2DP)	OS family and version; codec set; audio route policy	Model; firmware revision; codec set; sink capability	Stream the defined payload set for the defined duration	Streaming completes using the best jointly supported codec	Negotiated codec record; audio-gap event log	Proposed – not executed
INT-08 Audio control and telephony	OS family and version; declared control and telephony profiles	Model; firmware revision; declared control and telephony profiles	Exercise each declared control and telephony operation	Every declared operation produces the expected state change	Per-operation result log; peer state transitions	Proposed – not executed
INT-09 Data transfer	OS family and version; declared data transport	Model; firmware revision; declared data transport	Transfer the defined payload set with integrity verification	Every payload delivered intact within the window	Transfer log with per-payload integrity result	Proposed – not executed
INT-10 Reconnection and automatic re-establishment	OS family and version; background-execution policy; reconnection policy	Model; firmware revision; reconnection and advertising behaviour	Out-of-range and power-cycle recovery, with bond intact and after bond removal	Automatic re-establishment while bonded; successful re-pairing after bond removal	Reconnection trace with attempt count and elapsed time	Proposed – not executed
INT-11 Multi-	OS family and	Model;	Establish	All use cases	Per-link event	Proposed

connection and coexistence	version; concurrent-link capability and limit	firmware revision; concurrent-link behaviour	concurrent links to the defined accessory set and exercise each use case	complete with no link loss	streams; link-loss and priority-change log	– not executed
INT-12 Connection stability and environmental robustness	OS family and version; transport; power-control support	Model; firmware revision; power class	Hold the link under the defined load in each declared environmental condition	No unintended disconnection; timeliness within the scenario bound	Ambient channel-occupancy profile; connection-event log; latency samples	Proposed – not executed
INT-13 Error handling, timeout and recovery	OS family and version; platform error reporting; harness control surface	Model; firmware revision; declared failure behaviour	Inject each declared failure mode and measure recovery	Return to the pre-failure functional state within the scenario timeout	Failure-injection log; recovery timeline; fault state at end of run	Proposed – not executed
INT-14 Security and privacy validation	OS family and version; pairing policy; data-reachability policy	Model; firmware revision; pairing and key-strength support	Record negotiated method and key strength; verify encryption before application data; verify data reachability; test re-pairing	Strongest jointly supported method used; minimum key strength met; no exposure of sensitive payloads	Negotiated-method record; key-strength record; encryption-state trace; reachability test result	Proposed – not executed

6.4. Framework Traceability and Evaluation Criteria

Each test category in Table II is linked to at least one evidence source, and each metric in Section V-C is linked to the requirement identifiers whose outcomes it aggregates. Traceability is thus bi-directional: a reported metric breaks down into the requirement and pairing cells that produced it, and each of the cells breaks down into the events and sources that warranted its expected outcome. The evaluation criteria used for the framework are completeness for the connection life cycle, source-grounding of all the expected outcomes, reproducibility of all the metrics from recorded evidence, and separability of the four outcome classes.

6.5. Empirical Results

No empirical results are presented. No tests were conducted, test logs were not generated, and there is no measurement equipment involved in the creation of this article. Each quantitative value in the Table III is a value that has been reported by the cited source, with the context and limitations of that source as reported and is not a derived value from a combination of values from two different studies. The results in VI-A through VI-D are thus results of literature reviews and framework design, and are not experimental measurements of device performance. The framework is a validation methodology that a laboratory can execute; Section V-B states the device-selection and repetition requirements such an execution would need to satisfy.

7. Discussion

The framework addresses the research questions as follows. The six testing dimensions of Section III-C (capability intersection [4], security negotiation [5], [7], [12], platform policy [16]–[18] and environmental sensitivity [13], [14]) provide answers to RQ1 by detailing a mechanism. The four levels of separation of conformance, profile interoperability, end-to-end functional compatibility and user-perceived reliability, each with a different evidence requirement, and the outcome classification that does not lead to a report of a permitted fallback as a defect on RQ2. RQ3 is answered by the fourteen fully specified categories of Table II. RQ4 is answered by the traceability mapping of Section VI-C, which shows that the documented evidence base concentrates on security and link timing while leaving reconnection and multi-connection coexistence largely unevidenced. RQ5 is answered by the data-handling policy of Section V-D, which excludes indeterminate and unsupported runs from denominators while reporting them separately so that the denominator is always reconstructable.

Interoperability is a parameter of a device population, not a characteristic of a model, for handset manufacturers and OS vendors, because the negotiated configuration, the platform policy and the ambient environment will change during a single model's lifespan. For accessory developers, a profile version declaration is not enough to declare compatibility, as the size of the intersection of optional features in that version is sufficient. The number of runs that a test programme contains is not its

greatest asset to an interoperability lab; rather the value of the test programme comes from its coverage model and evidence chain, and the qualification vocabulary of waived and not-required verdicts should not be lost, as it is the distinction between a recorded non-testing and accidental omissions of a test [3]. For quality assurance teams, it is the execution and bookkeeping costs of interoperability testing that dominate, and this is precisely the cost that automation eliminates [15].

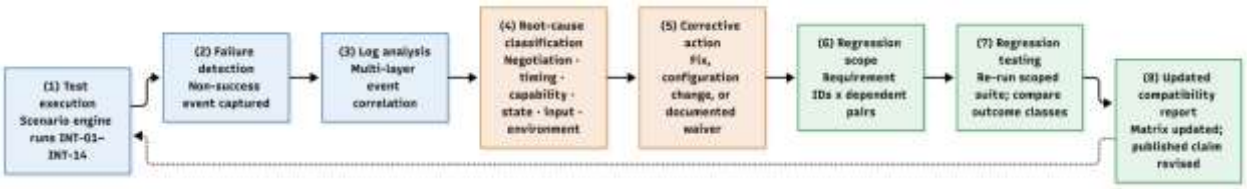


Figure 5. Closed-Loop Failure Diagnosis and Regression Testing, From Test Execution to Updated Compatibility Reporting

The current methods have a major drawback, which is not the absence of test cases, but the absence of a closure between detection and correction. If one pair has a defect, it is usually entered with the pair, while a mechanism that resulted in the defect is entered as a negotiation rule, a timing assumption, an input-validation gap or an environmental assumption is entered with a class of pairs. The closed loop the framework proposes is shown in Fig. 5: failures are created during execution, log analysis identifies the mechanisms that are responsible for the failures, the mechanism identifies the requirement identifiers and pairs that the corrective action must re-verify, and regression testing updates the compatibility matrix and published compatibility report, changing a test programme from a periodic audit to an on-going interoperability claim.

Two positions in this article are interpretations rather than verified findings. The first one is that the four-level separation of interoperability is the proper taxonomy—the design decision is one that has been documented and not established by experiment. The second is that the fourteen categories provide full coverage of the connection life cycle, completeness being argued against the life-cycle decomposition and the evidence base reviewed and a lab applying the framework might identify additional categories their product class needs. A second possible possibility that has not been investigated is that the mechanism-level classification of component 11 predicts the categories in which it is most likely to fail, when it is not yet tested; in this case, testing would require a campaign of the type described in Section V-B.

8. Threats to Validity and Limitations

The primary threat is source-selection bias. The evidence base was put together from a number of sources that were readily available in full text, which is likely to be more open-access publications, as well as vendor documentation and specification documents, whereas a number of other sources that appeared relevant were not available in full text so were not included but cited from their abstracts. Two specification documents were requested, but could not be found in full: the Core Specification version 5.2 (which was not received), and the feature description in Section II-A was obtained from the Bluetooth SIG feature overview [2] and the adopted specification register [4] only instead of the specification text; and one profile application note [23] was received, but not as a text document so only its subject matter was considered here, rather than its content. One survey [11] was checked based on its indexing record, but not on full text. Readers who need to have detailed information should read the specification directly.

All the empirical values in Table III are subject to device-sample limitations. The security studies were convenience samples of chips and devices, respectively, with the reported vulnerability percentages being proportions of the tested samples, not probabilities of the installed base [5], [7]; the fuzzing study involved twelve devices from eight vendors [8] and is also a convenience sample of the installed base and not a probability sample. This is exacerbated by hardware and firmware variation: the fuzzing results depend on the specific firmware versions that were examined [9] and a bug can be fixed in a firmware update that does not change the model designation so a hardware-specific result is limited by the firmware.

Measurement and environmental limitations affect the transferability of the performance values. The timeliness measurements were measured in one lab with a different hardware/stack configuration and a different payload type [14] and the discovery model is not measured directly by hardware, but by simulation [13]; thus, the framework does not assume that environment verification can be done optionally by adding an annotation, and it is a precondition. Another limitation is profile and version heterogeneity - multiple versions of each profile are in field use [4] and so a compatibility result for one profile version does not carry over to another; the coverage metric is set up specifically to ensure that this non-transferability is not hidden in an aggregate. The historical scope itself is a limitation: The technology baseline is for the 2020 period; later sources are only included as labelled context; no later capability is assigned to the 2020 technology baseline. Last, however, the framework itself has never been tested in its entirety and the completeness, definition of metrics, and classification scheme are proposals whose practical adequacy has not yet been demonstrated.

9. Conclusion and Future Work

In this article, the issue of how to assess Bluetooth interoperability in a heterogeneous handset and accessory market has been discussed. The evidence presented above demonstrates that failure of interoperability is not an uncommon occurrence and not limited to the security space: key negotiation and authentication are shown to be vulnerable to manipulation on the vast majority of tested devices [5], [7], [12]; LE system-on-chip implementations are vulnerable to key manipulation on a malformed device across multiple vendors [8], [9] and link timeliness is shown to degrade by five times under co-located interference [14]. It also leaves it open how to interpret the term conformance evidence to mean that conformance evidence does not guarantee interoperability, because it is caused by the intersection of two independently declared sets of features, as defined in (9).

The principal contribution is a thirteen-component framework comprised of fourteen fully-specified test categories, eight mathematically-defined metrics with declared observation windows and denominators, a four-way classification for test outcomes, distinguishing defects from degraded, unsupported and indeterminate outcomes, and a two-way test traceability scheme that connects the metrics to the requirement-and-pair cells as well as each requirement and pair cell to its evidence. The framework adds a vocabulary and a coverage model to a problem which is currently reported in incomparable terms, and clarifies the difference between requirements of a specification and the engineering criterion being proposed. The value of it is that it is a claim that can be made for a whole population of products, rather than a single product, and it can be audited on a population level and reconstructed from the denominator. The limitations are as given in Section VIII; it is a proposed methodology, it is based on a source base for which retrieval and sampling limitations apply, and it is not executed.

Four directions follow. First, an empirical campaign on a stratified device population satisfying the selection and repetition requirements of Section V-B, reporting all eight metrics with exclusion counts. Second, automated regression suites, extended to product to product runs to work with the platform harnesses as shown in profile testing [15]. Third, broader platform coverage because the policy differences found for the two dominant mobile platforms [16]–[18] imply results are platform-conditioned. Fourth, repeatable test environments with documented and controlled radio conditions, if otherwise timing results in various laboratories cannot be compared. The fifth, longer-term direction is the standardised of interoperability results reporting, in the form of the coverage matrix of Table IV, making the claims made by various organisations directly comparable.

References

[1] Bluetooth Special Interest Group, “Bluetooth Core Specification Version 5.2,” Bluetooth SIG, Kirkland, WA, USA, Dec. 2019. [Online]. Available: https://www.bluetooth.com/specifications/specs/

[2] M. Woolley, “Bluetooth Core 5.2 Feature Overview,” Bluetooth SIG, Jan. 2020, Version 1.0.0. [Online]. Available: https://www.bluetooth.com/wp-content/uploads/2020/01/Bluetooth_5.2_Feature_Overview.pdf

[3] Bluetooth Special Interest Group, “Qualification Program Reference Document (QPRD), Version 5,” Bluetooth SIG, Kirkland, WA, USA. [Online]. Available: https://www.bluetooth.com/wp-content/uploads/Files/Specification/HTML/QPRD/out/en/index-en.html

[4] Bluetooth Special Interest Group, “Specifications and documents: adopted specification register,” Bluetooth SIG, Kirkland, WA, USA. [Online]. Available: https://www.bluetooth.com/specifications/specs/

[5] D. Antonioli, N. O. Tippenhauer, and K. Rasmussen, “KNOB attack: Key negotiation of Bluetooth,” in Proc. 28th USENIX Security Symp., Santa Clara, CA, USA, Aug. 2019.

[6] Antonioli, D., Tippenhauer, N. O., & Rasmussen, K. (2019). The KNOB is broken: Exploiting low entropy in the encryption key negotiation of Bluetooth BR/EDR. Proceedings of the 28th USENIX Security Symposium (USENIX Security 19), 1047–1061. https://www.usenix.org/conference/usenixsecurity19/presentation/antonioli

[7] Sethi, M., Peltonen, A., & Aura, T. (2020). Misbinding attacks on secure device pairing and bootstrapping. Journal of Information Security and Applications, 51, Article 102461. https://doi.org/10.1016/j.jisa.2020.102461

[8] Ryan, M. (2013). Bluetooth: With low energy comes low security. Proceedings of the 7th USENIX Workshop on Offensive Technologies (WOOT ’13), 1–7. https://www.usenix.org/conference/woot13/workshop-program/presentation/ryan

[9] Sun, D.-Z., Sun, L., & Yang, Y. (2019). On secure simple pairing in Bluetooth standard v5.0 Part II: Privacy analysis and enhancement for low energy. Sensors, 19(15), Article 3259. https://doi.org/10.3390/s19153259

[10] Armis, “The dangers of Bluetooth implementations: Unveiling zero day vulnerabilities and security flaws in modern Bluetooth stacks (BlueBorne),” Armis White Paper, 2017. [Online]. Available: https://media.armis.com/pdfs/wp-blueborne-bluetooth-vulnerabilities-en.pdf

[11] Lonzetta, A. M., Cope, P., Campbell, J., Mohd, B. J., & Hayajneh, T. (2018). Security Vulnerabilities in Bluetooth Technology as Used in IoT. Journal of Sensor and Actuator Networks, 7, 28. https://doi.org/10.3390/jsan7030028

[12] Antonioli, D., Tippenhauer, N. O., & Rasmussen, K. B. (2019). The KNOB is Broken: Exploiting Low Entropy in the Encryption Key Negotiation Of Bluetooth BR/EDR. 28th USENIX Security Symposium (USENIX Security 19), 1047–1061.

[13] K. Cho, G. Park, W. Cho, J. Seo, and K. Han, "Modeling and performance analysis of device discovery in Bluetooth low energy networks," Sensors, vol. 15, no. 1, pp. 59–78, 2015, doi: 10.3390/s150100059.
[14] M. Spörk, C. A. Boano, and K. Römer, "Improving the timeliness of Bluetooth low energy in noisy RF environments," in Proc. Int. Conf. Embedded Wireless Systems and Networks (EWSN), 2019. [Online]. Available: https://www.carloalbertoboano.com/documents/spoerk19timeliness.pdf
[15] auto-pts contributors, "auto-pts: The Bluetooth PTS automation framework," GitHub repository. [Online]. Available: https://github.com/auto-pts/auto-pts
[16] Google, "Bluetooth overview," Android Developers Documentation. [Online]. Available: https://developer.android.com/develop/connectivity/bluetooth
[17] Google, "Bluetooth low energy overview," Android Developers Documentation. [Online]. Available: https://developer.android.com/develop/connectivity/bluetooth/ble/ble-overview
[18] Apple Inc., "Core Bluetooth programming guide: About Core Bluetooth," Apple Developer Documentation. [Online]. Available: https://developer.apple.com/library/archive/documentation/NetworkingInternetWeb/Conceptual/CoreBluetooth_concepts/AboutCoreBluetooth/Introduction.html
[19] Apple Inc., "What's new in Core Bluetooth," WWDC19 Session 901, 2019. [Online]. Available: https://developer.apple.com/videos/play/wwdc2019/901/
[20] Bluetooth Special Interest Group, "Bluetooth specifications and profiles: interoperability tools," Bluetooth SIG, Kirkland, WA, USA. [Online]. Available: https://www.bluetooth.com/specifications/interoperability-tools/
[21] Bluetooth Special Interest Group. (2019, May 6). 2019 Bluetooth® market update. https://www.bluetooth.com/ja-jp/bluetooth-resources/2019-bluetooth-market-update/
[22] European Commission DIGIT, "Interoperability test bed," Interoperable Europe. [Online]. Available: https://interoperable-europe.ec.europa.eu/collection/interoperability-test-bed-repository/solution/interoperability-test-bed
[23] Silicon Labs, "AN992: Hands-free and headset profiles," Silicon Labs Application Note. [Online]. Available: https://www.silabs.com/documents/public/application-notes/AN992.pdf