Emerging Paradigms in Wearable Security

Adaptable and Secure Sandboxing for On-the-Fly Collaboration Among Wearables

Diana A. Vasile | Nokia Bell Labs
Fahim Kawsar | Nokia Bell Labs and University of Glasgow
Chulhong Min | Nokia Bell Labs

We propose a novel security protocol for on-the-fly collaboration among wearables, addressing significant security challenges, such as data exposure and false information injection. Leveraging wearables' position on the body, our protocol ensures secure collaboration and enables new possibilities for ubiquitous computing.

Wearable devices are becoming increasingly versatile by enhancing their functionality through various sensors and interfaces. The advent of compact artificial intelligence (AI) accelerators, such as Analog MAX78000[1] and Google Coral Micro,[2] is making these devices smarter by enabling AI even in small wearable devices. Considering the exponential growth of wearables, we envision a new class of applications that leverage an on-the-fly collaboration of these wearable devices, as shown in Figure 1. This collaboration harnesses the collective strengths of the wearable ecosystem, enabling even simple devices to provide enriched services. For example, an application can seamlessly monitor a variety of health and activity levels by dynamically combining various sensors on distributed wearables, such as fitness bands, smartwatches, and hearing aids, based on their availability. Similarly, the application can provide alerts via different interfaces dynamically, e.g., voice feedback when earbuds are in use or haptic feedback when a smart ring is detected.

This new paradigm of on-the-fly collaboration among wearable devices enables the wearables to overcome their limitations in terms of resource scarcity and placement dependency, but also introduces significant security challenges and vulnerabilities. The interconnectedness brings together a variety of entities—including applications, machine learning models, and devices—that have access to users' privacy-sensitive data from different devices. The potential for malicious entities to exploit these vulnerabilities is substantial. Such entities could expose sensitive user data obtained

Digital Object Identifier 10.1109/MSEC.2024.3440198
Date of publication 21 August 2024; date of current version 11 November 2024.

from different devices for collaboration to unauthorized external parties[3] or inject false data into the collaborative network,[4] thereby compromising the integrity and effectiveness of these collaborative efforts.

Addressing the security challenges posed by on-the-fly collaboration is challenging due to several key issues. First, existing security protocols for wearable devices are primarily designed for a one-time association with smartphones, leaving them inadequately equipped to tackle runtime security problems that emerge from ongoing dynamic device collaboration. Second, as wearable devices evolve to become more compact and shed direct user interaction features, like screens and buttons, the complexity of implementing adaptive security measures increases. This trend toward miniaturization makes it more challenging to notify users of potential security issues or to engage them in the security process, further complicating the task of protecting user data in these densely interconnected ecosystems.

In this article, we propose a novel security protocol designed specifically for the collaboration of wearables. Our protocol aims to ensure adaptable and secure collaboration among wearable devices, safeguarding against the security threats that emerge in this innovative technological landscape. Through our protocol, we provide a framework for isolation application processes and enforcing access controls across multiple wearable devices when these collaborate. We also explore a method to autonomously allow devices to detect whether they are positioned on the same body at the present time or whether they produce valid data, which addresses a unique viewpoint of wearables: They are located directly on the user body and regularly have ready health-related biometrics.

Emerging On-the-Fly Wearable Collaboration

In the rapidly evolving landscape of wearable technology, we anticipate a shift toward on-the-fly wearable collaboration. This paradigm reshapes our interaction with the digital world, offering a more seamless, context-aware, and dynamic user experience. With an increasing proliferation of wearables, each equipped with a variety of sensors and interfaces, the potential for real-time, adaptive collaboration among devices is growing.

We envision this on-the-fly wearable collaboration to happen on multiple aspects: data, models, and functionalities. For instance, data coming from multiple wearables located in different parts of the body give a more rich and nuanced dataset to extract information. The selection of sensors can also be dynamically adjusted based on device availability and the expected accuracy.

The emergence of compact AI accelerators, such as Analog MAX78000 and Google Coral Micro, supports the move toward on-device AI processing, and also enables wearables to collaborate by distributing AI model processing tasks among themselves.[5] This distributed approach to AI, known as *model partitioning*, allows a set of wearables to meet the needs for concurrent models to run, especially when models have requirements extending beyond the abilities of a single wearable.

Functionalities can also be collaborated on; for instance, an application that monitors surrounding noises from the microphone on the smartwatch could notify the user through a vibration on the smart ring, or an audible announcement in the earbuds when the need for increased attention for the user arises in the environment.

Such collaboration enables wearable devices to overcome their limitations, both in terms of power and resource availability, but most importantly in the type of applications they can offer. With a collaboration effort, the wearables are no longer tied with a singular application (e.g., health–sleep tracking for a smart ring), but they can be part of a cluster of devices offering a bigger range of applications.

Threats

On-the-fly wearable collaboration offers exciting benefits, ranging from enhanced context-awareness to improved resource efficiency, but also raises the impact some security vulnerabilities have.

One straightforward concern is the potential for malicious entities within the collaboration network, which could mean either compromised devices, machine learning models, or applications. These entities pose a significant risk by potentially exposing sensitive user data collected across various devices to unauthorized external parties. Such a breach not only compromises user privacy but also undermines the integrity of the collaborative ecosystem.

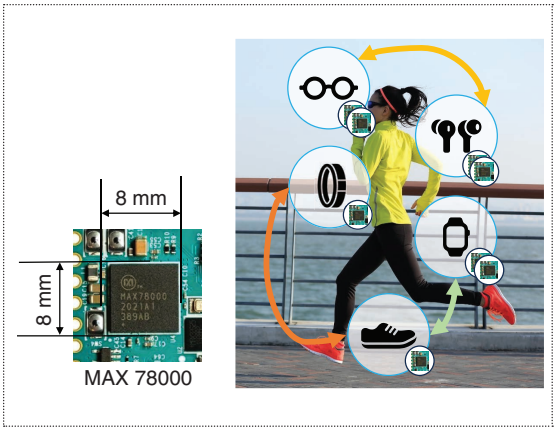

Figure 1. On-the-fly collaboration of wearable devices equipped with AI accelerators; each arrow represents a collaboration case.

Additionally, the dynamic and decentralized nature of this collaborative network increases the susceptibility to data integrity attacks, where false data could be injected into the system, leading to erroneous decision-making processes. This risk is particularly pronounced given the reliance on sensor fusion and AI-driven decision-making mechanisms that underpin these collaborative interactions.

Wearables are uniquely positioned on the human body to capture highly sensitive data about its wearer, ranging from health-related readings to granular location information, yet the lack of authentication and authorization leaves current wearable ecosystems vulnerable.[3,6] Furthermore, most wearables are currently able to only address one task or application at a time, so process isolation was not a high priority. However, with on-the-fly collaboration, more than one task or application could run on a wearable concurrently, strengthening the need for a sandboxing approach.

Application Sandboxing

Application sandboxing is a security mechanism used to isolate applications and prevent them from accessing certain resources or functionalities on a system, thereby reducing the potential damage that could result from malicious activities or software bugs.[7]

There are several sandboxing techniques, each designed to provide a different level of isolation and security depending on the requirements of the application or system.[8] Operating system (OS)-level sandboxing involves mechanisms—such as jails, chroot, and containers like Docker—which partition and restrict access to resources at the OS level. Virtualization-based sandboxing utilizes hypervisors and hardware-assisted virtualization to create isolated virtual machine environments. Browser sandboxing isolates web pages and plugins within separate processes or application programming interfaces to mitigate browser-based exploits. Mobile application sandboxing restricts each application's access to device resources and user data on mobile OSs through sandboxed environments and permission systems. Cloud-based sandboxing utilizes virtual private clouds and container orchestration tools to isolate and secure workloads in cloud environments. Network-based sandboxing employs virtualized network appliances and cloud-based sandbox services to analyze and detect malicious activities within network traffic.

We observe in practice that traditional application sandboxing techniques primarily focus on providing isolation within the confines of a single device,[9,10] a concept we refer to as *intradevice sandboxing*. These methods are designed to confine and control the execution of application processes within a single computing environment, effectively limiting their access to system resources and sensitive data to enhance security. However, as computing environments become increasingly interconnected and distributed, there emerges a need for more sophisticated approaches to ensure the secure operation of applications across multiple devices. Interdevice sandboxing involves distributing the sandboxing process across multiple devices within a network or ecosystem.

Interdevice sandboxing represents a paradigm shift in how applications are secured and isolated, as it extends the principles of sandboxing beyond the confines of individual devices to encompass entire networks or distributed systems. Unlike intradevice sandboxing, which focuses on isolating applications within a single device, interdevice sandboxing involves coordinating and orchestrating sandboxing mechanisms across multiple interconnected devices. This enables more comprehensive protection against security threats and enhances resilience in the face of evolving attack vectors.

One key difference to consider between intradevice and interdevice sandboxing is the scope of isolation and control. Intradevice sandboxing primarily addresses security concerns within the context of a single device, limiting the impact of malicious activities or software vulnerabilities on that particular device. In contrast, interdevice sandboxing extends this isolation across multiple devices, thereby mitigating the risk of lateral movement and propagation of threats within a network or distributed environment. This broader scope requires a more holistic approach to security, encompassing not only individual devices but also the interactions and dependencies between them.

Moreover, interdevice sandboxing introduces nontrivial challenges and considerations that differ from those encountered in traditional intradevice sandboxing. These challenges include ensuring consistent enforcement of security policies and access controls across heterogeneous devices, managing communication and synchronization between distributed sandboxes, and addressing scalability and efficiency in restricted environments. By leveraging the collective resources and intelligence of multiple wearable devices, we overcome the inherent limitations of these restricted devices, while creating protocols for interdevice sandboxing facilitates effective and secure sharing of resources.

A Protocol for Dynamic Interdevice AI Application Sandboxing

We envision that AI applications are deployed in a device-agnostic manner, such that the applications are not tied to running on specific hardware. We propose a novel protocol for interdevice application sandboxing that balances the access isolation of application

processes across multiple wearable devices, as well as tracks the restrictions of each individual device in the ecosystem.

Protocol Overview

In Figure 2 we provide an example run of the protocol components for an emergency user notification application (App X), which monitors when a user is mobile and unaware of their surroundings and there is imminent danger (e.g., the user is listening to music in voice isolation mode on the earbuds and a car is honking nearby). If such an event is detected, an audio or haptic notification should be provided urgently to the user. At the deployment stage, App X's developers create as part of the application specification two policies: the application access policy and the minimum requirements policy. The application access policy provides the necessary access control specification for the protocol to create the interdevice isolation requirements. The minimum requirements policy operates from an application needs perspective: it specifies what minimum requirements the application needs to operate effectively, such as inertial measurement unit (IMU) data, sound, and a user notification method.

When the application is installed for the user's ecosystem it uses these two policies to inform the type of sandboxing required. On App X's wake-up, the device on which the application was installed builds a device selection policy, which provides a clear definition of the type of devices the user owns that can run the application in an efficient and effective way. To build the device selection policy, the device uses the information known about the other devices the user owns, together with the two policies provided with the application. When the application needs to run, a dynamic subset of devices is selected that is best suited to meet the application's minimum requirements policy, such as creating an on-the-fly collaboration between a smartwatch, a pair of smart earbuds, and a smart ring. Prior to running, a set of validation strategies can be employed, which help ensure the correct runtime has been selected, such as verifying that the devices in the subset are currently on the same body. Finally, during operation the devices that might go offline are tracked to ensure that the application is able to finalize its run before the devices have gone offline. We detail each component in the "A Protocol for Dynamic Interdevice AI Application Sandboxing" section.

Application Deployment Stage: Specifying Policies

During the application deployment stage, the application developer establishes two policies necessary for secure and efficient operation within the wearable ecosystem.

The application access policy is a critical component to ensure the proper functioning of an application within a sandbox environment. This is because the sandbox must encompass all of the necessary components, including files and sensor data paths, required for executing the application securely. A fundamental consideration in defining access restrictions within the policy is to ensure that the sandbox does not grant the application more access permissions than necessary. Therefore, creating a policy with fine-grained control over the resources that the application process may utilize is essential to maintain security and minimize potential risks. We provide a list of permissions the application access policy tracks in Table 1. A notable complexity arises from the fact that the AI application may exhibit different behaviors and resource requirements across various deployment scenarios. For instance, in one deployment scenario, the application may primarily gather data from the IMU sensor embedded in a smart ring, complemented by audio input from earbuds. In contrast, in another scenario, the application may integrate IMU data from a chest band and audio input from a smartwatch's microphone. This variability underscores the importance of flexibility within the application access policy, allowing it to adapt and accommodate diverse deployment configurations while maintaining stringent access controls and security measures.

The minimum requirements policy is a second policy defined by the application developer. In the absence of strong ties to specific hardware, it specifies the minimum needs of the application for it to be able to execute. This policy is distributed with the application to inform the system whether it is able to meet the minimum requirements to allow the application to run collaboratively. Key components include: network connectivity, specifying the necessary Internet speed, reliability, and latency; power and battery resources, defining power consumption expectations and battery life requirements; memory and storage space, ensuring sufficient RAM and disk space for installation

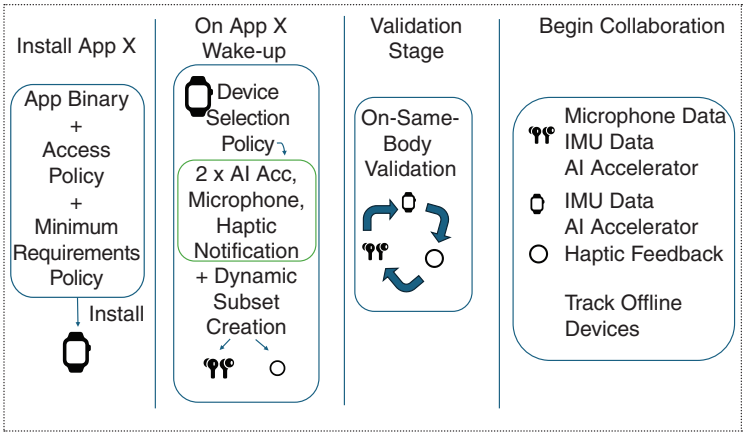

Figure 2. An example of secure AI collaboration workflow. IMU: inertial measurement unit.

and operation; sensor types and placements, identifying required sensors and their optimal locations for functionality and data collection; performance expectations, setting benchmarks for response times, throughput, and latency; environmental conditions, specifying any environmental factors, such as temperature or humidity that may impact operation; and security and privacy considerations, addressing encryption standards, authentication mechanisms, and data protection measures to safeguard user information and prevent unauthorized access. By detailing these components, the minimum requirements policy provides comprehensive guidance for ensuring compatibility, reliability, and optimal performance of the application across diverse collaboration environments.

Application Wake-up: Device Selection and Subset Creation

In the proposed dynamic environment, successive runs of the applications might require different preruntime preparations when devices change on the body as the opportunities for collaboration also change, which means that some further steps need to be performed in which we analyze what available devices there are and what is their current resource availability, prior to organizing them into a collaboration subset in which we can deploy the sandboxed application.

The device selection policy, established and managed within the user's environment, could either reside on the primary device, such as the smartphone, or on the device on which the application resides. This

Table 1. Types of permissions tracked by the access policy for AI applications in wearables.

Permission Type	Level of Access	Communication	Security Risks
Sensor access	Read access to sensors, such as accelerometer, gyroscope, and heart rate monitor.	Required for gathering user activity and health data.	Risk of unauthorized data collection or privacy infringement.
Audio input access	Read access to built-in microphones or external audio devices.	Used for voice commands, audio recordings, and sound analysis.	Risk of eavesdropping or recording sensitive conversations.
Camera access	Read/write access to built-in cameras or external camera peripherals.	Enables image and video capture, object detection, and augmented reality.	Risk of unauthorized surveillance or invasion of privacy.
Location data access	Read access to GPS or location tracking data.	Required for location-based services, navigation, and activity tracking.	Risk of location tracking and potential exposure of user's whereabouts.
Health data access	Read access to health-related data, such as heart rate, sleep patterns, and activity levels.	Facilitates health monitoring, fitness tracking, and personalized recommendations.	Risk of unauthorized access to sensitive health information.
Biometric data access	Read/write access to biometric data, such as fingerprint scans or facial recognition.	Used for authentication, identity verification, and personalized user experiences.	Risk of identity theft, biometric spoofing, or unauthorized access.
Network access	Read/write access to network resources for data transmission and communication.	Enables Internet connectivity, cloud services, and remote data access.	Risk of data interception, unauthorized access, or malware infiltration.
Storage access	Read/write access to local storage for saving and retrieving files and data.	Facilitates data caching, offline operation, and file management.	Risk of data loss, corruption, or unauthorized access to sensitive files.
Device control	Read/write access to device features, such as screen brightness, volume, and vibration.	Allows user interface customization, device settings adjustments, and hardware control.	Risk of device malfunction, unauthorized modifications, or disruption of user experience.
External device access	Read/write access to external devices, such as smart home appliances or Internet of Things devices.	Facilitates device integration, data exchange, and automation.	Risk of device tampering, data breaches, or unauthorized control.
Data encryption access	Read/write access to encryption keys and cryptographic functions.	Enables data encryption, decryption, and secure communication.	Risk of cryptographic vulnerabilities, key exposure, or data leakage.

policy is initialized during application installation and is influenced by the two policies defined in the deployment stage. Should the user introduce a new device to the ecosystem, the device selection policy undergoes updates accordingly. This policy tracks various factors, including device types, body locations where the wearables are worn, and available resources. It assesses potential collaboration scenarios among devices to optimize application performance. However, it is important to note that actual resource availability on wearables may vary during runtime due to factors, such as device load and concurrent operation with other applications. Therefore, the device selection policy adapts dynamically to account for these fluctuations and ensure efficient resource utilization.

Before creating the intradevice sandbox, the protocol needs a clear view of which devices are involved in the collaboration for the application. We therefore introduce the concept of subsets, to which some of the user's preassociated devices belong. To form a subset, the devices need reauthorization of each device to create a secured channel between each other. Discovery of devices and setup of this secured subset are essential steps in the protocol. Ensuring secure and dynamic authorization mechanisms accommodate the changing set of devices, preventing unauthorized access and promoting collaboration security. Once devices are associated with a subset (temporary association) a secure network needs to be bootstrapped to allow for a secure collaboration. There are several steps involved in the network creation. If there is a network already established to which one of the two devices belongs, then the device should invite the newly added device to the subset onto the network. If there is not already a network established, then the two devices should proceed to establish a network between themselves.

Upon Collaboration Request: Validation Strategies

With every dynamic iteration, several validation strategies can be employed to ensure the application is in a healthy state to run.

The on-same-body verification is specific to the wearable ecosystem. Within this environment, we restrict the applications to run only on the devices that are currently worn by the same user. Since wearables are portable devices, the user could easily share with another person: for example, if they are sharing a pair of earbuds to listen to music together. As such, before every application run is deployed, the wearables have to agree they are on the same body, reading the same context information.

Another validation strategy is to validate the authenticity of the applications to prevent the execution of unauthorized or malicious software. One approach is to utilize common techniques, such as digital signatures, application attestation mechanisms, and application reputation services to verify the origin and integrity of applications before allowing them to run.

It is essential to verify the integrity of each participating device before allowing it to join the sandboxing ecosystem. Where available, apply techniques, such as secure boot, code signing, or designing an alternative integrity measurement to ensure that devices have not been compromised or tampered with.

Collaboration in Progress: Monitoring Components and Changes

During the application runtime, several approaches can be employed, such as a fully autonomous run in which the application deploys its tasks to the wearables and expects that eventually they will return the result, or a tracked operation, in which the application tracks the resources to which it deployed the tasks. In the former, we need a designated way to handle devices going offline and therefore not delivering the task it was sent, while in the latter we can assume a resource management tracker could be deployed to ensure the tasks and devices are monitored.

In a dynamic wearable environment, devices can go offline due to limitations on power resources or interruptions in connectivity. When a device goes offline, it may fail to deliver the task assigned to it, leading to potential disruptions in the application workflow. To address this challenge, a comprehensive strategy for handling offline devices is required. To run the application tasks in a fully autonomous run, which ensures the devices do not have to waste resources on recurrent tasks, such as sending periodic heartbeat signals or performing network status checks, we propose an adaptive estimation of device liveness. Additionally, fault tolerance mechanisms, such as task replication or retry policies, can mitigate the impact of device failures and ensure task completion, even in the presence of intermittent connectivity or device outages.

A resource management tracker plays a crucial role in optimizing resource utilization and ensuring efficient operation of the application during runtime. In this instance the collaboration between wearable devices for an application can be monitored by a more powerful device, such as a smartphone. The tracker monitors various resources across distributed wearables, including CPU usage, memory utilization, battery levels, and network bandwidth. By continuously monitoring resource availability and utilization patterns, the tracker dynamically allocates tasks to devices with adequate resources and balances the workload to prevent resource exhaustion or bottlenecks. Additionally, the resource management tracker facilitates proactive resource management by identifying potential performance bottlenecks or resource constraints in real time and triggering

appropriate remedial actions, such as task migration or load balancing. Furthermore, the tracker provides valuable insights into resource usage trends and performance metrics, enabling optimization strategies and capacity planning for future autonomous runs in the same environment.

Establishment of a Collaboration Network: Interdevice Communication

During a dynamic subset creation stage, the device on which the application resides, such as the smartwatch (SW), initiates a request for connection to the devices included in the device selection policy. Each request for connection contains the originating device's identifier (SW_ID), the application access policy details, which also includes a list of required components, and a new shared secret for this collaboration (SK_x), all encrypted with the most recent session shared secret (SK).

Should the devices accept the connection request, they send back their current bio-IDs (see next section for details of how this bio-ID is generated), encrypted with SK_x. If the bio-IDs from the three devices match, then smartwatch initiates the collaboration, each device proceeds to provide the functions they agreed to, and in the case of our example application, should a risk to the user be identified, the smart ring can notify the user through a haptic feedback (Figure 3).

A Case for On-Same-Body Verification

We introduce an innovative concept, time-bound contextual bio-IDs,[11] to support on-same-body verification for wearable devices, particularly when these devices have no capability for password input, e.g., touch screen. These IDs are universal representations of sensor data embedded in a common latent space, ensuring individuality across different users and contexts. More specifically, we generate a time-bound bio-ID from vital signs, such as heart rate, oxygen saturation, and blood pressure. These metrics are inherently universal for an individual regardless of the device placement, but unique according to the individual's dynamic and contextual factors. By comparing these bio-IDs, the on-same-body validity can be dynamically verified, eliminating the need for additional model training for user authentication. For example, our proposed protocol declines collaboration requests if different wearables generate disparate bio-IDs, which may indicate that the devices are worn by different users or that malicious entities have produced falsified data.

A straightforward way to generate bio-IDs would be to extract and combine device/sensor-specific features or absolute values of various vital signs. However, sensor readings or features from these devices exhibit variability due to heterogeneity in device location, hardware specifications, and software characteristics. To generate robust bio-IDs, we leverage the contrastive learning technique. Contrastive learning is a machine learning approach that aims to understand patterns by distinguishing between similar and dissimilar data points. It works by comparing pairs of data points: positive pairs, which are expected to be similar, and negative pairs, which are expected to be dissimilar. This method

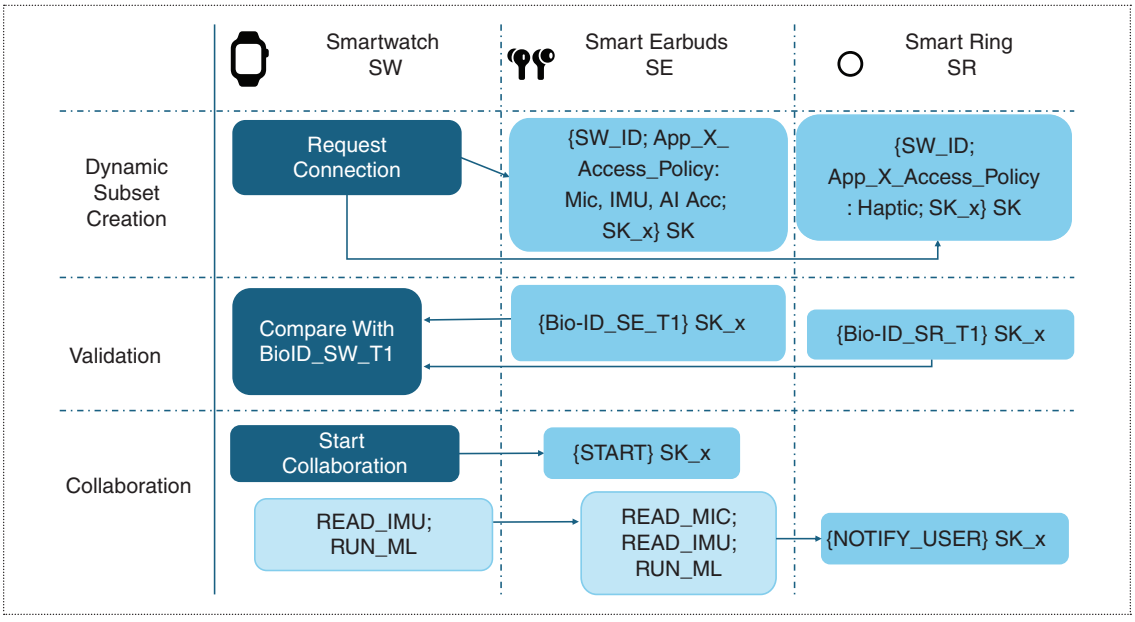

Figure 3. Example of protocol secure communication between three devices collaborating. SW: smart watch; SE: smart earbuds; SR: smart ring.

trains a model to bring embeddings of positive pairs closer together in a latent space, while pushing embeddings of negative pairs further apart.

In our context, positive pairs with sensor data coming from multiple wearables worn by the same user at the same time encourage embeddings to become more similar, while data from different users or different time points act as negative pairs, promoting distinct embeddings. The embedded representations are structured in the form of 1D arrays and are further processed for matching purposes. Even when the raw data do not show correlation, the resulting embeddings are expected to be aligned when the devices are on the same body at the same time, and misaligned when the data were captured from the same body but at different times, or from different bodies. Figure 4 presents an example of a photoplethysmography (PPG) signal and the corresponding embeddings on two earbuds for the same user versus different users. The results demonstrate that our bio-IDs are time-bound, meaning the embeddings differ even for the same user when generated at different times.

The proposed system to enable bio-ID generation and matching works uses a three-stage process: predeployment stage, upon deployment, and runtime bio-ID generation and matching. In the predeployment phase, the focus is on training bio-ID models to function efficiently across various wearable devices and sensors, balancing model complexity with runtime accuracy. Our strategy is to train models based on device placement rather than individual devices and create embedding models for sensor combinations to ensure adaptability and robustness in dynamic device contexts.

Upon deployment, we update the models to include new user data by extracting time-synchronized sensor values from various devices for positive samples, potentially requesting users to wear all devices for validation. Negative samples are created using misaligned data from device pairs and a global dataset to ensure a diverse set of negative candidates.

At runtime, we monitor available devices, generate bio-IDs, and match them using a lightweight model with three fully connected layers for the encoder, two for the projection head, rectified linear unit activation, and 10% dropout for robustness. The user matching model seeks to identify if two samples originate from the same user and time window, employing a similar architecture with three fully connected layers and binary output for classification. The goal is for the embeddings to be consistent for the same user across different devices at the same time, while ensuring distinctiveness for different users or times. For energy-efficient continuous matching, we employ local change detection to trigger bio-ID matching only when significant changes are detected, optimizing battery usage while maintaining accuracy.

Our evaluation with the FatigueSet dataset[12] across multiple device placements, users, and activities has shown that we can process these embeddings with such high quality that, even when the raw data do not show any correlation, the resulting embeddings are aligned when the devices are on the same body at the same time. When the bio-IDs are generated with sensor readings from the same user, the average Spearman's rank correlation coefficient between bio-IDs is 0.75 with a standard deviation of 0.20. However, when the bio-IDs are generated with sensor readings from different users and compared, the average coefficient is 0.12 with a standard deviation of 0.31.

We further evaluated our proposed system on Raspberry Pi 4B and Raspberry Pi Zero W to measure system cost. The system demonstrates low latency, CPU usage, and energy consumption, supporting nine and 6.4 operations per second, respectively. The main latency arises from sensor-specific operations, not from the system itself, and while bio-ID generation is CPU-intensive, its overall impact is minimal due to short latency periods.

Discussion

Among validation strategies, as an initial attempt we developed and demonstrated on-same-body verification—using the concept of bio-IDs to compare whether two devices are placed on the same human

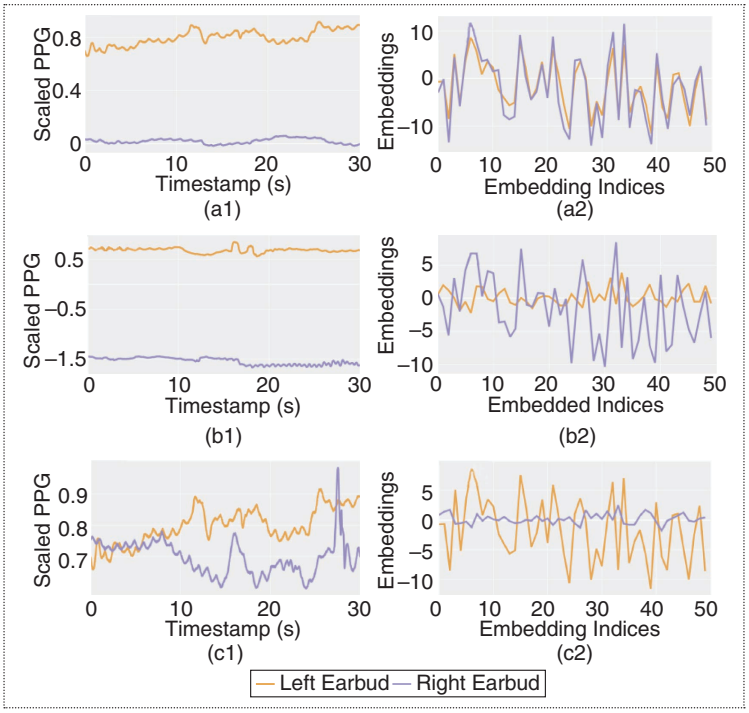

Figure 4. Example of PPG (a) and embeddings (b) on two earbuds. (A) Same user, time-aligned; (B) different users; (C) same user, nontime-aligned. Excerpted from Orzikulova et al.[11]

body and produce valid data—because it can be universally applied to various collaboration scenarios. In contrast, other validation strategies, such as application authenticity checks and device integrity verification, need to be specifically designed and crafted by considering the collaboration requirements of applications and device specifications, respectively. We will delve deeper into the collaboration scenarios and develop these strategies in future work.

Looking beyond, we envision that on-the-fly wearable collaboration will expand from a single-user device ecosystem to a multiparty device ecosystem, encompassing other users' devices and nearby Internet-of-Things (IoT) devices. This expansion not only amplifies the utility of AI applications but also introduces a higher order of security vulnerabilities. With the inclusion of a diverse array of devices, such as smartphones, smartwatches, fitness trackers, and IoT devices, in collaborative ecosystems, the attack surface significantly expands, increasing the potential avenues for exploitation by malicious actors. For instance, the proliferation of sensor-equipped wearables and IoT devices opens up opportunities for unauthorized access to sensitive user data, such as health metrics, location information, and personal preferences. Additionally, the interconnectivity between devices and the broader network infrastructure raises concerns about data interception and manipulation, where attackers may eavesdrop on communication channels or tamper with transmitted data to compromise the integrity of collaborative processes. Furthermore, the heterogeneity of devices introduces challenges in maintaining consistent security configurations and enforcing access controls, leading to potential misconfigurations or policy conflicts that could be exploited by adversaries. Overall, while expanding the wearable collaboration offers unprecedented opportunities for innovation and efficiency, it also underscores the critical importance of robust security measures to mitigate the associated risks and safeguard the integrity and confidentiality of data exchanged within these ecosystems. It does not only necessitate stronger requirements for existing security mechanisms, such as secure computation and trusted execution environments, but sandboxing also remains critical to ensure data isolation while facilitating robust and safe collaboration.

We look to explore the application of interdevice sandboxing in these broader scenarios, focusing on developing protocols that support secure collaboration across a wider variety of devices without compromising the autonomy or security of individual devices. More specifically, since collaboration requirements and policies could conflict due to the different objectives of various users and the smart environment, new policies and validation strategies need to be designed. Furthermore, the protocol also needs to be adapted to resolve these conflicts. By addressing these challenges, devices can seamlessly and securely collaborate across boundaries, unlocking new possibilities for ubiquitous computing and smart environments.

We discussed a solution to provide on-the-fly secure wearable device AI collaboration by formalizing a dynamic interdevice application sandboxing protocol. By providing a framework for isolating application processes and enforcing access controls across multiple devices, the protocol ensures the integrity and confidentiality of data exchanged within collaborative ecosystems. Through the exploration of the on-same-body verification validation strategy the protocol addresses key security challenges associated with multiparty device collaboration. However, the other validation strategies—application authenticity checks and device integrity verification—remain unaddressed. Furthermore, the expansion of wearable collaboration to encompass a broader range of devices introduces new complexities and security vulnerabilities that must be addressed in future research. Future work will focus on developing protocols that support secure collaboration across diverse device ecosystems, while also considering the conflicting collaboration requirements and policies of different users and environments. By addressing these challenges, we can unlock the full potential of ubiquitous computing and smart environments, while also ensuring the privacy, security, and autonomy of individual users and devices. ■

References

1. "MAX78000." Analog Devices. Accessed: Aug. 15, 2024. [Online]. Available: https://www.analog.com/en/products/max78000.html
2. "Dev board micro." coral.ai. Accessed: Aug. 15, 2024. [Online]. Available: https://coral.ai/products/dev-board-micro/
3. R. Unuchek. "How I hacked my smart bracelet." Securelist. Accessed: Aug. 15, 2024. [Online]. Available: https://securelist.com/how-i-hacked-my-smart-bracelet/69369/
4. J. Xin, V. V. Phoha, and A. Salekin, "Combating false data injection attacks on human-centric sensing applications," in *Proc. ACM Interactive Mobile Wearable Ubiquitous Technol.*, Jul. 2022, vol. 6, no. 2, pp. 1–22, doi: 10.1145/3534577.
5. T. Gong, S. Y. Jang, U. G. Acer, F. Kawsar, and C. Min, "Collaborative inference via dynamic composition of tiny AI accelerators on MCUs," 2023, *arXiv:2401.08637*.

6. F. Blow, Y. Hu, and M. Hoppa, "A study on vulnerabilities and threats to wearable devices," *J. Colloq. Inf. Syst. Secur. Educ.*, vol. 7, no. 1, p. 7, 2020.
7. V. Prevelakis and D. Spinellis, "Sandboxing applications," in *Proc. FREENIX Track USENIX Annu. Tech. Conf.*, Boston, MA, USA, Jun. 2001, pp. 119–126.
8. F. Al Ameiri and K. Salah, "Evaluation of popular application sandboxing," in *Proc. Int. Conf. Internet Technol. Secured Trans.*, Piscataway, NJ, USA: IEEE Press, 2011, pp. 358–362.
9. T. Dunlap, W. Enck, and B. Reaves, "A study of application sandbox policies in Linux," in *Proc. 27th ACM Symp. Access Control Models Technol.*, 2022, pp. 19–30, doi: 10.1145/3532105.3535016.
10. C. Greamo and A. Ghosh, "Sandboxing and virtualization: Modern tools for combating malware," *IEEE Security Privacy*, vol. 9, no. 2, pp. 79–82, Mar./Apr. 2011, doi: 10.1109/MSP.2011.36.
11. A. Orzikulova, D. A. Vasile, F. Kawsar, and C. Min, "Time-bound contextual bio-id generation for minimalist wearables," 2024, *arXiv:2403.00889*.
12. M. Kalanadhabhatta, C. Min, A. Montanari, and F. Kawsar, "FatigueSet: A multi-modal dataset for modeling mental fatigue and fatigability," in *Proc. Int. Conf. Pervasive Comput. Technol. Healthcare*, 2021, pp 204–217.

Diana A. Vasile is a research scientist at Nokia Bell Labs, CB3 0FA Cambridge, U.K. Her research interests include trust establishment, transparency, and privacy. Vasile received a Ph.D. in computer science from the University of Cambridge. She is a Member of IEEE. Contact her at diana-alexandra.vasile@nokia-bell-labs.com.

Fahim Kawsar leads the pervasive systems research at Nokia Bell Labs, CB3 0FA Cambridge, U.K., and holds a professorship at University of Glasgow, G12 8QQ Glasgow, U.K. His research interests include earables, sensory systems, Internet of Things, and computational behavior modeling. Kaswar received a Ph.D. in computer science from Waseda University. He is a Member of IEEE. Contact him at fahim.kawsar@nokia-bell-labs.com.

Chulhong Min is a principal research scientist and tech lead at Nokia Bell Labs, CB3 0FA Cambridge, U.K. His research interests include on-device artificial intelligence, embedded systems, and Internet of Things. Min received a Ph.D. in computer science from the Korea Advanced Institute of Science and Technology, South Korea. He is a Member of IEEE. Contact him at chulhong.min@nokia-bell-labs.com.