HoloPair: Securing Shared Augmented Reality Using Microsoft HoloLens

Anonymous Author(s)

ABSTRACT

Augmented Reality (AR) devices continuously scan their environment in order to naturally overlay virtual objects onto user's view of the physical world. In contrast to Virtual Reality, where one's environment is fully replaced with a virtual one, one of AR's *"killer features"* is co-located collaboration, in which multiple users interact with the same combination of virtual and real objects. Microsoft recently released HoloLens, the first consumer-ready augmented reality headset that needs no outside markers to achieve precise inside-out spatial mapping, which allows centimeter-scale hologram positioning.

However, despite many applications published on the Windows Mixed Reality platform that rely on direct communication between AR devices, there currently exists no implementation or achievable proposal for secure direct pairing of two unassociated headsets. As augmented reality gets into mainstream, this omission exposes current and future users to a range of avoidable attacks. In order to close this real-world gap in both theory and engineering practice, in this paper we design and evaluate *HoloPair*, a system for secure and usable pairing of two AR headsets.

We propose a pairing protocol and build a working prototype to experimentally evaluate its security guarantees, usability, and system performance. By running a user study with a total of 22 participants, we show that the system achieves high rates of attack detection, short pairing times, and a high average usability score. Moreover, in order to make an immediate impact on the wider developer community, we have published the full implementation and source code of our prototype, which is currently under consideration to be included in the official HoloLens development toolkit.

ACM Reference format:
Anonymous Author(s). 2016. *HoloPair*: Securing Shared Augmented Reality
Using Microsoft HoloLens. In *Proceedings of 2017 Annual Computer Security Applications Conference, USA, PR,* 13 pages.
DOI: 10.1145/nnnnnnn.nnnnnnn

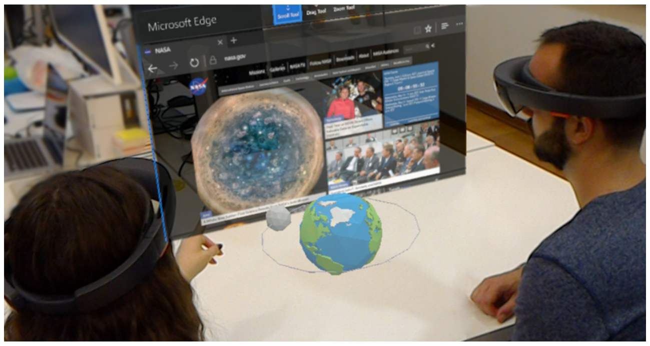


Figure 1: A real-world view of a shared augmented reality experience of two of our study participants. In this paper, we focus on the challenge of securely and usably pairing augmented reality headsets in order to support a multitude of collaborative AR use cases and applications.

1 INTRODUCTION

Augmented Reality (AR) technologies allow overlaying virtual objects over one's real-time view of the physical world. In contrast to virtual reality, in which user's environment is completely replaced with a virtual one, AR enabled devices (such as headsets, mobile-phones, car windshields, etc.) continuously monitor and sense their surrounding in order to create an experience in which the virtual content is naturally blended with the real world.

Many of the largest tech companies are placing AR into focus of their near- and mid-term plans. Recent examples include Facebook making AR the central topic of their 2017 F8 developer conference, Apple developing their own AR spectacles and recruiting top experts from the field [23], and Google investing more than $540 million into an AR startup Magic Leap [7], while also working on Tango [11], their own AR computing platform.

Windows Mixed Reality platform. Microsoft recently released HoloLens, a self-contained (fully untethered) head mounted computer, the first publicly available headset that supports the wider Windows Mixed Reality (WMR) platform, which has already been integrated into the core Windows 10 functionality[1].

HoloLens's main innovation is its remarkably precise inside-out position tracking that does not rely on any outside sensors or markers. This, in turn, enables developers to render realistic 3D virtual objects (holograms) that are precisely *anchored*

[1]Recent announcement from the company stated that, by the end of 2017, *"an update to Windows 10 will enable mainstream PCs to run the Windows Holographic shell and associated mixed reality and universal Windows applications."* [10]

in the environment and interact with existing physical objects (as shown in Figure 1). Consequently, Microsoft brands HoloLens as a "Mixed Reality" device, emphasizing the anchoring capabilities over the simpler, HUD-style overlays, that most existing AR devices support. [2]. While HoloLens is currently the only commercial device with such capabilities, this is rapidly changing, as manufacturers such as Acer, ASUS, Dell, HP, and Lenovo [27] all recently announced their headsets for the WMR platform.

Shared Augmented Reality. Besides individual use, HoloLens headsets open a wide range of possibilities for collaboration of co-located users, in which they all experience precisely the same virtual objects embedded in their shared environment. One such example is NASA Jet Propulsion Laboratory's who already *"allow their scientists to work on Mars"* by using HoloLens devices to train, plan, and execute their Mars rover missions [17]. Other examples include doctors using HoloLens during spinal chord and brain surgeries [20], or Israel and US defense forces using the device for training and mission planning [1, 4].

At the core of many of these security- and privacy-critical applications is the ability to directly connect multiple headsets in order to show the same, precisely located holograms to multiple users. However, when two previously unassociated devices establish a connection for the first time, it is crucial to prevent potential man-in-the-middle (MITM) attacks by relying on an independent out-of-band channel [15].

Unfortunately, the only published research on the topic of secure pairing of AR headsets heavily relies on wireless localization capabilities, which are currently not available in any consumer-facing AR device [9]. Consequently, despite the large number of published AR applications, the quickly growing WMR Platform currently lacks any implementation or practical research proposal to securely perform direct pairing of two or more AR headsets.

The Challenge. Despite the extensive previous research on secure device pairing, which we overview in Section 9, existing protocols are not directly applicable to AR headsets since they fail to address two specific challenges. Firstly, instead of assuming a single user who controls two devices, AR pairing necessarily involves two users, each with their own headset. Since AR headsets are designed to be continuously worn throughout the course of a normal workday, users must not be required to take them off, and can, consequently, only observe the output from their own device. Secondly, considering the proliferation of AR devices, each of which include multiple front-facing cameras, it is necessary to assume that the adversary can fully eavesdrop the out-of-band communication. This is not a typical adversary model considered in device pairing, where the adversary usually only controls the wireless network.

Contributions. We address this challenge by relying on the unobservability of AR displays and by exploiting HoloLens' state-of-the-art inside-out positioning. This allows us to design protocols that are based on precisely positioned shared holograms to augment users' communication channels, while at the same time retaining the usability of the system.

In this paper, we design *HoloPair*, a practical and usable system that achieves secure pairing of two augmented reality headsets. In order to evaluate its security guarantees, usability, and performance, we implement a working prototype of the system using two Microsoft HoloLens devices, and run a comprehensive user study with 22 participants. As the measurements show, the majority of participants are able to successfully detect simulated attacks or confirm that pairing was successful in as little as 8 seconds. Despite participants' lack of earlier experience with AR headsets, the results of the post-experiment questionnaire show high subjective opinion of the usability of the *HoloPair* system.

Finally, as the number of devices that support the WMR platform grows, the developer community will likely continue publishing new applications that rely on shared augmented reality experiences with increasing speed. In order to ensure that security concerns are considered from early stages of the growth of the developer community, we've made the full implementation available to the public and started the process to include the source code in Microsoft's official HoloLens-Unity repository.

2 WHAT'S DIFFERENT ABOUT HOLOLENS?

Spatial mapping and Inside-out tracking. The main innovation of Microsoft HoloLens over previously available augmented reality headsets is considered to be precise inside-out position tracking and spatial mapping of the surrounding environment. The inputs from two *"environment understanding cameras"* on each side, a depth camera, an ambient light sensor, and a 2MP video camera are combined in the custom made *Holographic Processing Unit* (HPU) to build and maintain a model of the surrounding objects, which are then used as anchors for precise head and object tracking.

This allows developers to create immersive mixed reality experiences in which the position of virtual objects (holograms) is fixed in space with centimeter scale precision, and they naturally react to collisions with the physical world or remain at their location over multiple sessions or even months.

Sharing holograms: *World Anchor.* Shared holographic experiences, in which the same holograms are shown on multiple devices in a specific physical space is at the core of many augmented current and future reality experiences, which range from 3D modeling, augmenting or collaborating during manufacturing, surgical, or mechanical procedures, to gaming or even military training.

Microsoft's current development toolkit (HoloToolkit-Unity [12]) provides support for such shared experiences in a form of *world anchors*. One of the devices establishes an anchor at a specific location and communicates its position within its model of the room to all other devices. The anchor is

[2]The debate about the appropriate use of the terms *"augmented reality"* and *"mixed reality"* is still ongoing. In this paper, we use the terms interchangeably.

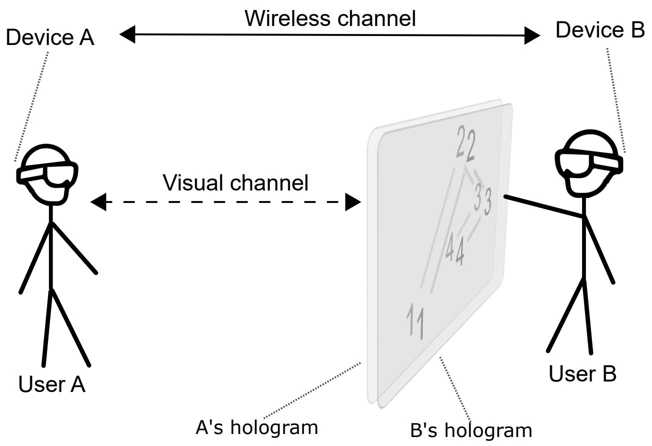


Figure 2: System model. Two individuals, each equipped with an AR headset, wish to establish a secure channel for subsequent communication. Devices communicate over the high-bandwidth wireless channel, which is controlled by the adversary. Users communicate over low-bandwidth visual channel, which can be eavesdropped, but not intruded by the adversary, since that would be detected by legitimate protocol participants. Each headset can independently and unobservably overlay arbitrary content over their user's view of the physical world.

then used as a reference point for the coordinate system, in which the shared holographic objects are located at the same position to each user.

Gesture Tracking. Using its multiple depth perception cameras, HoloLens is able to precisely track users' hands and recognize hand gestures such as *pointing & clicking*, zooming, or *bloom* (used as a generic "back/menu" command). Due to the lack of keyboard or mouse input, the gesture recognition module is the main control channel, used in different application to naturally manipulate, control, and even model or draw 3D objects in space.

3 ASSUMPTIONS AND GOALS

We assume a scenario in which two users, $\mathbf{U}_A$ and $\mathbf{U}_B$, that do not have a pre-shared secret and are both wearing an augmented reality headset (such as Microsoft HoloLens), meet at in person and want to securely connect their devices to share an AR experience.

3.1 System Model

As shown in Figure 2, $\mathbf{U}_A$ and $\mathbf{U}_B$ are each equipped with a trusted device, $\mathbf{D}_A$ and $\mathbf{D}_B$, which can augment their view of the physical world by independently drawing precisely positioned holograms.

We assume that the AR headsets can communicate over a high-bandwidth **wireless channel** and have no other direct channel of communication. Users, however, can communicate over an out-of-band audio or **visual channel** on which their headsets can independently and unobservably overlay content. Consequently, a way to view the system model from the perspective of the headsets is to imagine that they are using human participants as output channels for their communication over the out-of-band channels.

In our system model, we aim to not rely on the audio channel, as it is inherently undirected, polluted by each additional participant, and easily injected into. On the other hand, the visual channel does not depend on the environment noise, can be used in many scenarios where silence is expected (e.g. during lectures or meetings, in public spaces), and is significantly harder to undetectably inject into [6].

We assume that users do not have access to any other trusted third-party service that they could use to establish the connection, such as PKI infrastructure. Finally, since human participants are part of the protocol, we assume that each user will be involved in only one protocol run at any given moment.

3.2 Adversary Model

We assume a Doley-Yao style network adversary $\mathbf{E}$, whose goal is to use his device(s) to prevent $\mathbf{U}_A$ and $\mathbf{U}_B$ from establishing a secure connection, for instance by positioning himself as the man-in-the-middle.

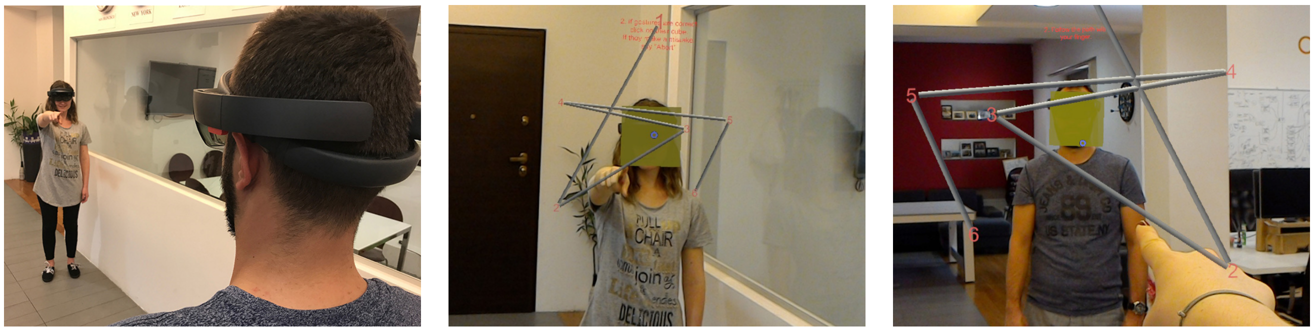
In contrast to most previous work on device pairing, we consider the adversary to be co-located and able to position himself arbitrarily close to one or both protocol participants. While this gives him the ability to fully eavesdrop on the visual channel, he must, however, remain passive, since any intrusion would be detected as suspicious behaviour by the legitimate protocol participants. For instance, if the protocol requires $\mathbf{U}_B$ to make a gesture, $\mathbf{E}$ is unable to prevent this from happening ("jamming" it) since that would be detected by $\mathbf{U}_A$, and would cause the protocol to abort.

However, he can not observe the independently generated holograms that each device overlays over their user's view of the physical world. We do not consider denial-of-service attacks.

3.3 Design Goals

Based on the discussion in the previous sections, we state the design goals for a successful system for pairing of augmented reality devices as:

- **Security.** The attacker must have a low chance of a successful man-in-the-middle attack. Users should detect attempts of a man-in-the-middle attack in the majority of cases.
- **Usability.** The system must not require users to take their headsets off. The interaction should be relatively short and users should be willing to perform it whenever they wish to share an AR experience with a new user/device. Most pairing attempts should result in a successful key confirmation.
- **Hardware Requirements.** The system should not require capabilities that are not available on current devices (specifically MS HoloLens). In order to allow seamless execution whenever two AR users decide to



(a) Users as seen outside of AR. **(b) View of User A.** **(c) View of User B.**

Figure 3: Views of both users as they are using the *HoloPair* system to pair their devices. $\mathbf{U}_B$ needs to follow the generated object with their hand, while $\mathbf{U}_A$ observes the hand movements and confirms that they indeed correspond to the holographic object generated on his $\mathbf{D}_A$ from the value $H_W(K|P_{\mathtt{KA}}|P_{\mathtt{KB}})$.

share their augmented realirtyis, the proposed pairing system should neither have high computational, memory, nor energy requirements.

4 THE *HOLOPAIR* SYSTEM

We now present *HoloPair*, a system that achieves usable and secure authenticated key exchange of two augmented reality headsets without relying on any third party.

4.1 System Overview

Our system builds upon the general idea of establishing a secure communication channel using Short Authenticated Strings [25], which we simplify and adapt to the specific usability considerations of AR headsets: having two users that should not be required to not take their devices off their heads, an an adversary who can observe all their actions.

The devices first exchange their public keys over the high-bandwidth, but insecure channel, then commit and agree on a specific instance of a weak-hash. Finally, the human participants confirm the authenticity of the exchanged keys using the low-bandwidth visual channel whose integrity is guaranteed.

Role of human protocol participants. In order to prevent man-in-the-middle attacks, the system relies on device owners to establish an out-of-band communication channel and authenticate the exchanged keys. During this manual process, they are aided by their AR headsets, thus allowing them to seamlessly confirm relatively high entropy in comparison to previous approaches.

4.2 Pairing Protocol

At the end of a successful protocol run, two previously unfamiliar devices ($\mathbf{D}_A$ and $\mathbf{D}_B$) should have exchanged and authenticated a pair of public keys, which can subsequently be used to bootstrap secure communication, for instance by deriving a shared symmetric key. The protocol, shown in Figure 4, consists of the initial device discovery and parameter agreement, followed by the three main steps:

0. Prerequisites. $\mathbf{U}_A$ initiates the protocol by broadcasting the willingness to pair and distributing the public parameters of the underlying cryptographic functions, together with the location of the *WorldAnchor*. Any other device that receives the broadcast will assume the role of $\mathbf{D}_B$ as soon as their user confirms the willingness to pair.

1. Public key exchange. $\mathbf{U}_A$ sends their public key $P_{\mathtt{KA}}$ and $\mathbf{D}_B$ responds by sending the public key $P_{\mathtt{KB}}$ over the wireless channel.

2. Weak-Hash Commitment. Since the out-of-band visual channel has low bandwidth, using it to directly compare and authenticate the exchanged keys would require unusably long time. Therefore, human participants are usually required to compare values (strings, images, colors, shapes) generated from different *weak-hash* functions, which have significantly smaller output entropy, and thus a larger probability of hash collision.

In order to prevent a man-in-the middle attacker from performing an off-line collision attack on the weak-hash by finding a suitable pair ($P_{\mathtt{KA}}$', $P_{\mathtt{KB}}$'), $\mathbf{U}_A$ commits to a specific instance of the weak-hash, defined by randomly chosen value K and sends its hash to $\mathbf{U}_B$. After $\mathbf{D}_B$ receives the commitment H(K), $\mathbf{U}_B$ is instructed to acknowledge the receipt over the visual channel (for instance by waving to $\mathbf{U}_A$), after which $\mathbf{U}_A$ opens his commitment by sending the encrypted value of K to $\mathbf{U}_B$. We further discuss the need for weak-hash commitment and visual acknowledgment in the security analysis in Section 5.

The encrypted message also includes the *WorldAnchor*, which specifies the origins of the shared coordinate system and the transformations between two devices.

3. Shared Secret Confirmation. After exchanging the value K, both devices can now independently compute the weak-hashes from the received public keys, $H_W(K|P_{\mathtt{KA}}|P_{\mathtt{KB}})$, which will be identical only if the exchanged keys are indeed authentic.

The reason for user participation in the protocol is to confirm that the exchanged public keys are authentic, and consequently, that the weak-hashes independently computed

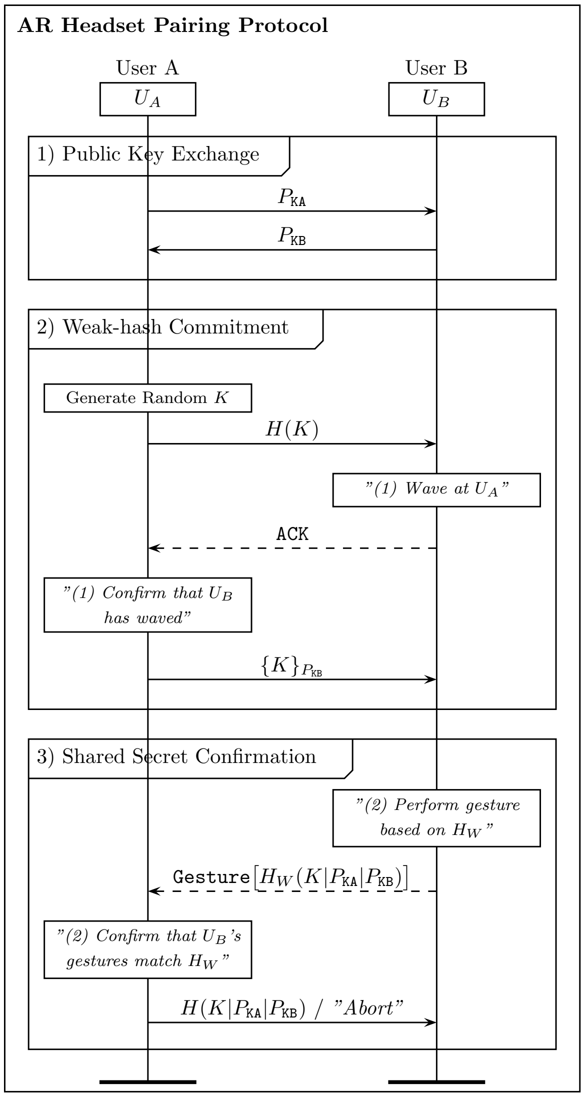

AR Headset Pairing Protocol
User A
U_A
User B
U_B
1) Public Key Exchange
P_{KA}
P_{KB}
2) Weak-hash Commitment
Generate Random K
$H(K)$
"(1) Wave at U_A"
ACK
"(1) Confirm that U_B has waved"
$\{K\}_{P_{KB}}$
3) Shared Secret Confirmation
"(2) Perform gesture based on H_W"
Gesture$[H_W(K|P_{KA}|P_{KB})]$
"(2) Confirm that U_B's gestures match H_W"
$H(K|P_{KA}|P_{KB})$ / "Abort"

Figure 4: The protocol consists of three main steps. (1) Using the insecure channel, the devices exchange their public keys. (2) Next, device A commits to a specific instance of the weak-hash H_W and privately opens it after $\mathbf{U}_B$ acknowledges receipt. (3) Finally, $\mathbf{U}_B$ uses the low-bandwidth visual channel (dashed lines) to communicate the weak-hash H_{W_B}, which $\mathbf{U}_A$ verifies and confirms/aborts the execution in the last message.

from their values indeed match. By relying on the unique capabilities of AR headsets, we increase the usability of the comparison by guiding users with holographic objects shown in space between them, which are generated to uniquely encode the value of the weak-hash values computed on their headsets.

Depending on the characteristics of the generated hologram, $\mathbf{U}_B$ is required to perform a specific gesture, while $\mathbf{U}_A$ confirms that the observed gesture matches his expectation. Finally, if $\mathbf{U}_A$ concludes that the shared key confirmation was successful, $\mathbf{D}_A$ sends the full hash $H(K|P_{KA}|P_{KB})$, which $\mathbf{D}_B$ uses to confirm that the pairing was indeed successful and displays the message to $\mathbf{U}_B$.

4.3 Gesture for Shared Secret Confirmation

We emphasize here that our protocol does not depend on the specific gesture (or some other procedure) that users use to confirm that they indeed share the same secret value $K|P_{KA}|P_{KB}$. Consequently, while designing and developing the *HoloPair* system, we implemented and tested several different versions of the shared secret confirmation, which we fully describe in Appendix B.

We base the remainder of this paper on the shared secret confirmation scheme that was shown to be the best performing, both in terms of theoretical security guarantees and subjective usability in our pilot user study (Section 7.1). As depicted in Figure 2 and visible in Figure 3, each independently computed weak-hash is used to construct a holographic shape that consists of N positions on a plane in the physical space between users. Given the exchanged *WorldAnchor* and the precise positioning, both AR headsets show the holographic shapes at exactly the same location. In order to verify that the constructed shapes match, $\mathbf{U}_A$ observes (on the visual channel) as $\mathbf{U}_B$ moves their finger along the generated shape and thus confirms that the weak-hashes generated on both devices indeed match, which is only possible if the exchanged public keys and the value K are authentic.

5 SECURITY ANALYSIS

We now evaluate the design of the *HoloPair* system according to the security goals from Section 3.3. First, we analyse the protocol's guarantees in a short security sketch in Section 5.1 and then discuss the likelihood of a successful random hash collision attack in Section 5.2.

5.1 Security Sketch

The attacker succeeds if he is able to eavesdrop on the communication between A and B after the protocol execution. Given that we base out protocol on existing proposals for establishing secure communication that are based on comparison of short secrets, we refer to earlier work for an extensive security proof [25] and provide an overview of the arguments in several claims:

Claim 1. In order to successfully eavesdrop on subsequence communication, the adversary must force legitimate users $\mathbf{U}_A$ and $\mathbf{U}_B$ to agree on a different set of public keys than they originally envisioned, P_{KA}', P_{KB}' without aborting the protocol execution. Otherwise, protocol participants would assume that the exchanged keys are not safe to use and likely repeat the protocol.

Claim 2. Since the attacker can not remain undetected if he tries intruding the out-of-band holographic channel in the last step, he must ensure that the out-of-band shared secret confirmation finishes successfully from both participants' point of view. Assuming attentive users (we discuss this assumption later), this can happen only if the weak hashes computed by both $\mathbf{D}_A$ and $\mathbf{D}_B$ are equal, namely that $H_{W_A}(K|P_{\mathtt{KA}}|P'_{\mathtt{KB}}) = H_{W_B}(K'|P'_{\mathtt{KA}}|P_{\mathtt{KB}})$.

Claim 3. Even though the entropy of H_W is not large, the attacker who tries impersonating $\mathbf{U}_A$ to $\mathbf{U}_B$ is required to commit to some value K' in Step 2, before discovering the value K (actually chosen by $\mathbf{U}_A$) in Step 3. This prevents the attacker from being able to perform an extensive off-line search for some suitable K' in the smaller output space of potential values of H_W that would result in a weak-hash collision. As a result, the likelihood of choosing such K' that results in exactly the right combination of $P_{\mathtt{KA}}$', $P_{\mathtt{KB}}$', and K' is inversely proportional to the output space of H_W. We emphasize that the MITM attacker could only learn the plaintext value of K send by $\mathbf{U}_A$ after already committing to some values $P_{\mathtt{KA}}$' and K' towards $\mathbf{U}_B$. Consequently, the attacker has no means of finding alternative values $P_{\mathtt{KA}}$' and $P_{\mathtt{KB}}$' whose resulting holograms appear more similar to human participants than any randomly chosen pair of weak-hashes.

Claim 4. Despite the initial exchange of the *WorldAnchor* over the insecure channel, its modification does not help the attacker since it can only cause a linear translation of $\mathbf{U}_B$'s holograms in the physical space. This would result in a mismatch between the positioning of holograms shown to $\mathbf{U}_A$ and $\mathbf{U}_B$, and cause one of the participants to abort the protocol.

In **conclusion**, the attacker whose goal is to position himself as a passive man-in-the-middle between $\mathbf{D}_A$ and $\mathbf{D}_B$ during their initial pairing has no better choice than trying to randomly guess a pair of replacement keys $P_{\mathtt{KA}}$', $P_{\mathtt{KB}}$' that would yield the same hashes H_W on both devices. Consequently, his chance of success is inversely proportional to the entropy of the output space of the weak-hash H_W, which directly depends on the chosen variant of the holographic shared secret confirmation. We analyze this probability in the next section.

5.2 Probability of a Weak-hash Collision

We now analyze the likelihood that a different pair of keys still results in a weak-hash collision $H_{W_A}(K|P_{\mathtt{KA}}|P'_{\mathtt{KB}}) = H_{W_B}(K'|P'_{\mathtt{KA}}|P_{\mathtt{KB}})$ for the shared secret version of the confirmation step described in Section 4.3. We analyze the other two variants of the shared secret confirmation in Appendix B.

The shape of each possible instance of the shared hologram is uniquely determined by its N coordinate pairs (X_i, Y_i). In our implementation, we use a total of 10 different values for both X_i and Y_i, which results in a probability that another pair of keys results in exactly the same shared hologram to be:

$$P(N) = \frac{1}{(10 \times 10)^N} = \frac{1}{100^N}$$

We note here that due to headsets' holographic guidance, the theoretical entropy of the shared secret confirmation step is significantly larger than, for instance if users would be required to reading a sequence of strings or digits of a given number, as has been proposed in previous work. This additionally confirms the usability benefits that mixed reality devices can offer to many existing systems and security schemes.

Finally, by adapting the length of the sequence (defined by N) each of the variants of the confirmation step can be adapted based on the security needs and expectations of a specific scenario.

5.3 User Inattentiveness

Given the high output entropy of the used gestures for shared secret confirmation, it is likely that the most probable reason of attack success is user inattentiveness, which results in users not verifying the sequence carefully, or even immediately clickin*"Accept Gesture"* before any gesture was made by $\mathbf{U}_B$.

The problem of user attentiveness is a challenging one, both in terms of performance evaluation, and in terms of designing interfaces that would encourage one to pay attention that has received wide interest from the research community [24].

In order to give an estimate of the ability of *HoloPair* users to detect potential attacks, in the next section describe a working prototype of the *HoloPair* system. We use the prototype to run a user study with 22 participants in which we simulate a man-in-the-middle attack in 20% of pairing attempts to experimentally evaluate the security guarantees of the *HoloPair* system.

6 SYSTEM PROTOTYPE

In order to experimentally evaluate the feasibility, security guarantees, and performance of the proposed *HoloPair* system, we build a working prototype using two Microsoft's HoloLens devices and we make the source code and the implementation available to the public.

6.1 Source Code and Development

The prototype is written in the C# programming language, using the Unity framework [13]. When building the functionality specific to HoloLens, we rely the components from Microsoft's official HoloToolkit-Unity repository, which provides functionality such as spatial mapping, world anchors and gesture recognition.

HoloToolkit-Unity is a public repository on GitHub, with many contributions (merged pull-requests) coming from the wider developer community. We thus created a fork of the official repository, and packaged our prototype as one of the provided examples according to Microsoft's instructions. Excluding external references, our prototype consists of 3241 lines of C# code, which are located in the `Assets/Examples/HoloPair` folder.

The source code is available online. Since the motivation behind our work was not only to suggest a suitable pairing protocol, but also to improve the current security practices of the Windows Mixed Reality platform, we have started the process to have our code included into the official

HoloToolkit-Unity repository. Furthermore, the source code of the prototype implementation is publicly available at:

https://tinyurl.com/holopair

Building and Contributing. In order to build and run *HoloPair*, one should clone the repository, load the main HoloToolkit project in Unity, and open the `HoloPair` scene. After creating a Visual Studio solution from Unity, the solution should be deployed on two HoloLens devices connected on the same wireless network. The first device that loads the application will assume the role of $\mathbf{D}_A$, while the other will assume the role of $\mathbf{D}_B$.

We have made our best-effort to make the code readable and easily extensible for further development. Since we plan to continue actively developing the *HoloPair* prototype, we will gladly accept any comments, suggestions, or pull requests.

6.2 Main Implementation Components

We now briefly discuss the implementation of the main components.

Networking and device discovery. We use Unity's High-Level Networking API to discover other devices that are running the *HoloPair* prototype by broadcasting/listening to a specific message on port 8888. While our current prototype assumes that devices share the same wireless network, there is no limitation to extend the prototype and support direct ad-hoc wireless connections in the future.

Cryptographic functions. We use the standard Microsoft's implementations of the 2048-bit RSA PKCS1 for asymmetric cryptography and 256-bit SHA2 for hashing.

Constructing the shared hologram from the weak-hash. The shapes that represent *weak-hashes* are generated by extracting bits from the base-64 string representation of the full hash in order to generate N coordinates that define them. In our implementation, for each of N points that comprise the shape, we extract sufficient number of binary bits to generate one of 10 different X coordinates and one of 10 different Y coordinates.

Hologram sharing - establishing the shared *world anchor*. In order for multiple devices to show the identically located content to their users, they must first agree on a shared coordinate system that will be used as a frame of reference regardless of users' subsequent movements. In our prototype, we use HoloToolkit's implementation of *"World Anchoring"*, which in most cases achieves positioning errors smaller than a few centimeters.

Positioning of the shared holograms. In our current implementation, the shared hologram is shown on a line between users, initially 1.65 m from $\mathbf{U}_B$, and then moves towards the $\mathbf{U}_B$ during a period of 3 seconds, to finish at a distance of 0.85 m. We've made the design decision to implement such movement in order to ameliorate the slightly limited field of view of the current version of HoloLens ($30° \times 17.5°$). This allows $\mathbf{U}_B$ to first get the full view of the shape, and then to be close enough so that they can reach it with their hands.

Confirming protocol success & aborting. In the current prototype, $\mathbf{U}_A$ confirms that the observed gesture was correct by *gazing* at $\mathbf{U}_B$'s head and performing a *click* gesture, for which we use HoloLens' gesture recognition module. We deliberately use a gesture instead of voice commands to prevent the attacker from making an attack successful by simply generating, potentially even inconspicuously [6], a confirmation voice command.

However, we believed that it would be more convenient to use voice recognition for the case when users suspect to have detected an attack attempt. In such cases, we asked users to say *"Abort"*, expecting to increase usability over the *gaze and click* gesture. We discuss this (false) intuition in further detail in Section 8.

6.3 User Experience

As shown in Figure 3, despite the seemingly large number of protocol steps, *HoloPair* users are required to perform only two manual steps in order to securely pair their AR headsets.

Instructions for the user in role $\mathbf{U}_A$:

(1) Once you see $\mathbf{U}_B$ waving, gaze and click on them.
(2) Watch as $\mathbf{U}_B$ moves their finger along the shown shape. If their hand movement follows the shape, gaze and click on them. Otherwise say *"Abort"*.

Instructions for the user in role $\mathbf{U}_B$:

(1) Wave towards $\mathbf{U}_A$.
(2) Move your finger along the shape shown in front of you, starting from number 1.

As the usability evaluation in the next section shows, the manual steps are based on natural gesture that users learn quickly. While a few initially needed to practice the *gaze and click* gesture, this is a standard primitive that AR headset users are likely to already have mastered before starting to use *HoloPair*. However, the manual behavior that *HoloPair* introduces, namely following the shape with their finger, was naturally and easily performed by all protocol participants.

7 EXPERIMENTAL EVALUATION

We now experimentally evaluate the *HoloPair* system with respect to average pairing time, users' ability to detect man-in-the-middle attempts, and subjective usability, measured with a questionnaire. Additionally, we provide data on computational performance of the developed *HoloPair* prototype.

7.1 Pilot User Study

While designing and prototyping the *HoloPair* system, we implemented three different versions of the shared secret confirmation step (shown in Figure 9), and we chose the most suitable one after running a small-scale pilot user study.

In the pilot user study, 6 participants were instructed to repeatedly pair their headsets using the supported schemes in both roles, and were afterwards asked to state which scheme they found the most usable. Somewhat surprisingly, all

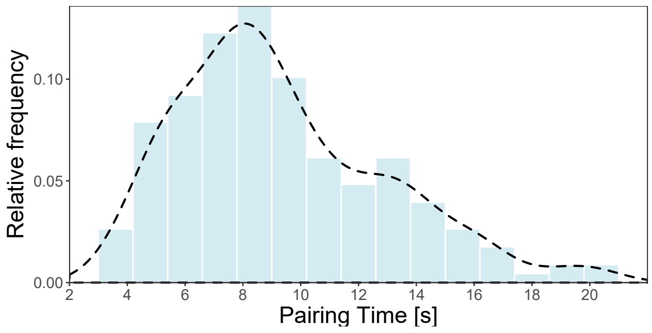


Figure 5: Relative frequencies of all pairing times in our main user study when no attack was simulated. Pairing times are computed from the moment from when $\mathbf{U}_A$ confirms that user $\mathbf{U}_B$ has waved until $\mathbf{U}_A$ has either confirmed a successful pairing or one of them said *"Abort"*.

participants indicated that they preferred the version of the shared secret confirmation step in which $\mathbf{U}_B$ is required to follow the shape generated in space with their hand (version **(c)** in Figure 9). Besides being subjectively the most usable, this scheme conveniently supports the highest theoretical entropy of the weak-hash ($(10 \times 10)^N$), and was also the easiest to explain to pilot-study participants.

Consequently, we decided to fully focus the main user study, which consisted of 22 new participants, on evaluating the performance, security, and usability of this variant of the shared secret confirmation.

7.2 Main User Study

In our main user study, we invited a total of 22 participants to experimentally use the system.

Demographics. We recruited a total of 22 participants (14 female, 18 male, aged from 21 to 33) using mailing lists and social media posts. None of the study participants owned a HoloLens device nor had extensive prior experience using AR headsets.

Setup. Participants were invited to the study in pairs, and were not grouped by any specific criteria except available times to participate in the experiment. Upon arriving, participants were introduced to goals of the experiment, given the chance to ask questions and asked to sign the consent form. The experiment introduction explained the reasoning behind the need to securely pair augmented reality headsets, the envisioned usage scenario and the potential attacks that can happen during the process. Participants were told that their goal will be to repeat the pairing attempt several times, and that we might simulate an attack during some of the attempts.

Procedure. Not having any previous experience with AR, study participants were first given the opportunity to get accustomed with using an augmented reality headset: specifically using gesture recognition (*click* to confirm a successful pairing), and voice recognition (say *"Abort"*when an attack is detected).

Each pair of participants was asked to perform a minimum of 10 pairing attempts, after which the protocol roles ($\mathbf{U}_A$, $\mathbf{U}_B$) were switched and participants performed at least additional 10 or more pairing attempts.

The experiments measured the impact of two independent variables:

(1) Attack Simulation. The main need for user involvement in the *HoloPair* system is to detect potential man-in-the-middle attacks, which are evident by a mismatch of independently generated weak-hashes on two headsets. In order to evaluate users' ability to detect such attacks, we simulated differing shapes being shown to participants in randomly chosen 20% of the pairing attempts.

(2) N, Shape Complexity. In order to evaluate the impact of complexity of the shared secret confirmation step on the dependent variables (total time and success rates), we varied the value of N, the number of shape segments. In each pairing attempt, N is randomly chosen from the set $\{4, 6, 8\}$.

Measured Data. After having a total of 22 participants take part in the main study, we gathered data on the execution of a total of 230 pairing attempts, out of which a man-in-the-middle attack was simulated in 44 cases.

For each pairing attempt, we measured two sets of dependent variables: **(1)** timestamps at which users entered each step of the pairing protocol, and **(2)** whether the pairing attempt was successful, either by detecting a potential attack (when it was simulated), or correctly exchanging the shared secret (when no attack simulation took place).

Given the non-sensitive data that we measured and stored anonymously, our institution does not require an Institutional Review Board approval for these kinds of studies. However, all study participants were given a written explanation of the goals of the study, signed a consent form, and were aware that their data will be used for publication.

Results: Pairing Time. Figure 5 shows the relative distribution of the total times for all pairing attempts in our main user study, measured from the moment when $\mathbf{U}_A$ reveals its commitment on a specific instance of the weak-hash. The median pairing time for users is **between 8 and 9 seconds**, while 80% of successful pairing attempts finish in less than 13 seconds. These times are comparable or close to previously reported confirmation times for similar device pairing schemes [2, 8]. Furthermore, considering that this process needs to be repeated only once for each new pair or devices, as we show in the remainder of this section, the majority of study participants found the System both usable and sufficiently fast.

It is interesting to note that we observed longer average pairing times in the case when users decided to abort the pairing execution, as shown in Figure 6. This is likely due to two reasons. Firstly, after observing that the gesture did not match the expectation, participants often repeat it before deciding to abort. Secondly, we observed that, despite the simplicity of the voice command (*"Abort"*), their instruction was sometimes not recognized by the device on the first

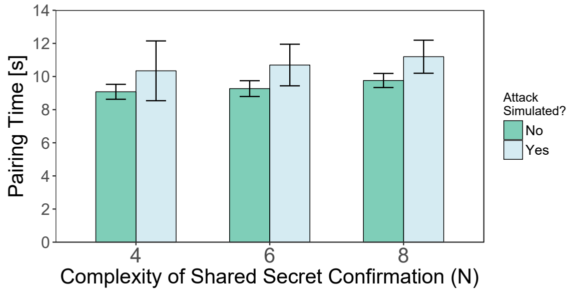


Figure 6: Average pairing times and standard error of the mean as a function of complexity of the shared secret confirmation step, N. As N changes from 4 to 8, the average pairing time increases, but only slightly. This indicates that the total time spent performing the gesture does not depend on its complexity as much as it does on other user behavior, such as waiting, making the decision, or *clicking*.

attempt, which increased the time before the device recorded that a decision was made.

However, we emphasize that in the case of attacks, the time until users make an ultimate decision is of significantly smaller influence. It is much more important that participants in our study detected potential attacks with high success rates, which we discuss next.

Results: Success Rates of Pairing and Attack Detection. Previous research has shown that users are often inattentive, do not understand the risks, or simply proceed without even trying to verify the exchanged shared secrets [24]. Since we simulate attack attempts in a percentage of pairing runs, we are able to estimate the likelihood of a successful man-in-the-middle attack even when the two weak-hashes do not collide, but users fail to recognize this. We consider a pairing attempt successful if $\mathbf{U}_A$ confirmed that shapes match when there is no attack simulation, or if the same participant called *"Abort"* in the case where an attack was indeed simulated.

The success rates that our system achieved in our user study are highly encouraging: our results show that **91% of simulated attacks (43/47) were detected** by the study participants. While there numbers are high, it is important to note the possibility that study participants in general are generally more vigilant by the virtue of being measured and performing a novel interaction. However, even though participants might become less attentive as they get accustomed to using the *HoloPair* system, we note here that the measured success rate is comparable or better than the results achieved in studies which similarly tested user's ability to compare different short strings, hashes, or pictures [15, 24].

An even higher success rate was achieved in the case where there was no attack, where **98% of pairing attempts were successful (181/186)**, with only 4 false aborts when both weak-hashes did indeed match. This further confirm that relying on precisely located holograms that are independently shown to both participants and using gestures to validate that their shapes indeed match allows for confirmation of fairly high-entropy information.

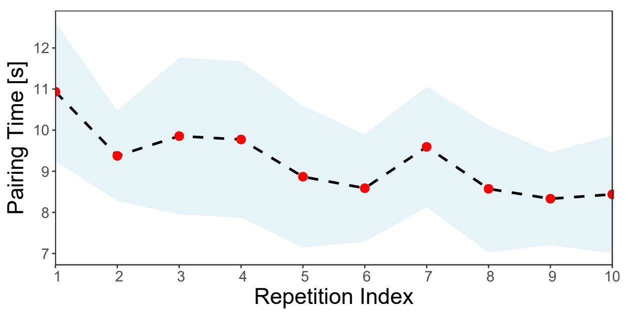


Figure 7: The impact of learning on the mean pairing times. As users repeat the same procedure multiple times, their pairing times decrease by about 20%.

Results: Impact of the Shared Hologram Complexity (N). We now look into how the the average augmented reality headset pairing times depend on N, the complexity of the secret shared hologram.

As shown in Figure 6, increasing N does expectedly increase the average required time for two users to pair their headsets, but only to an extent that is within 1 second. The small difference is visible both in the case of attack simulations (light green) and in the case when no attack was simulated (green).

The small difference in pairing times indicates that users spent the majority of pairing in other behavior, such as waiting, deciding, or inputing the decision into their own device (confirming via a *click* gesture, or aborting by saying *"Abort"*).

Results: Learning Effects.

Finally, we analyze the extent to which users learn to use the *HoloPair* system more efficiently with repetition of the pairing procedure. Figure 7 shows the mean and standard deviation of all successful pairing attempts in our study, grouped by their session index. All pairing attempts in which one participant takes the of $\mathbf{U}_A$ are considered a single session, and we thus have two sessions per participant pair.

As expected, the mean pairing time reduces as users repeat the procedure multiple times, from about 11 seconds in their first measured attempt, to less then 9 seconds on their 10th attempts (we do not take into account the "practice" attempts here). These results suggest that despite their initial inexperience with using AR headsets, participants indeed quickly become accustomed with the *HoloPair* system, as they also indicated in the usability questionnaire (Q7), which we discuss next.

7.3 Usability Questionnaire

After participants performed multiple pairing attempts in both roles, we asses the usability and user perception towards the *HoloPair* system and the implemented prototype by asking them to complete the System Usability Scale [5] (SUS)

SUS is a reliable and general tool for evaluating the usability of various systems, which has been widely used since its introduction [3]. The questionnaire consists of 10 statements such as *"Q6: I thought there was too much inconsistency in this system"* or *"Q9: I felt confident using the system"* that users of the system grade on a Likert scale (1 - Strongly disagree, 5 - Strongly agree). The full list of questions is provided in the Appendix A.

Results: Usability Questionnaire. We show the overall results of the SUS questionnaire in Figure 8. None of the study participants thought that they had to learn a lot before they could get going with the system (Q10), and while some felt they would need a help of a technical person (Q4), most users generally found the system easy to use (Q3) and believed others would learn to use the system very quickly (Q7).

While some users believed that they would need support of a technical person to be able to use the system, this is likely due to the fact that learning how to use the gestures did indeed require initial help from the experimenters, but was quickly grasped by the participants (as also visible in Figure 7). Furthermore, once users get accustomed to such gestures by using other Microsoft Holographic applications, they are likely to not need any help to start using the *HoloPair* system.

Based on the total of 22 participants who completed the questionnaire, none of which have had extensive previous experience using HoloLens, the *HoloPair* system achieves an **average overall SUS score of 86.9**. Previous research on interpreting individual scores concluded that the mean SUS score of 85 translates into users' adjective rating of "Excellent" [3]. Consequently, we conclude that the majority of the study participants found the usability of the *HoloPair* system to be well above the average.

7.4 Prototype Performance

Table 1 provides measurements for the total, average and maximum/minimum values of the following measurements: battery load, RAM, CPU and GPU load, and network bandwidth.

The maximal increase in GPU, RAM, and CPU load is bellow 15% of their maximal amounts, while the total network bandwidth for the full execution of the pairing protocol is about 700kB. Such small performance footprint is expected, considering the small number of messages that our system actually exchanges over the wireless network, and contrasting this with the need to exchange precise spatial data in the process of finding a *world anchor* for hologram sharing.

While performance was not one of our considerations during the development of the *HoloPair* prototype, and thus the measured values could likely be further improved, we conclude that the current prototype already achieves a low overall impact on the existing HoloLens device.

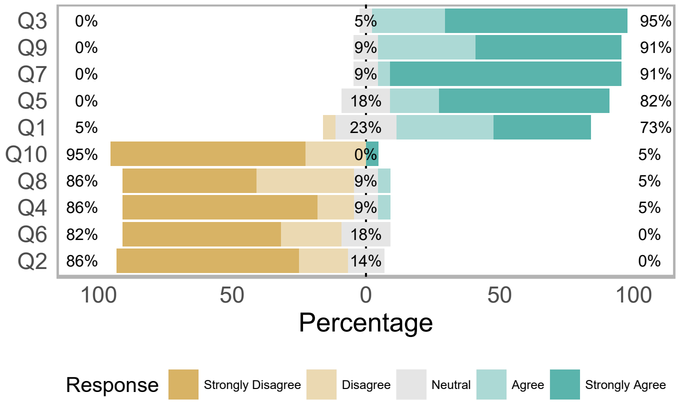


Figure 8: Participants' responses to the SUS questionnaire show a high average SUS score of 86.4. None of the participants thought that they had to learn a lot before they could get going with the system (Q10). While some felt they would need a help of a technical person (Q4), most generally found the system easy to use (Q3) and believed others would learn to use the system very quickly (Q7).

8 DISCUSSION

Automating the Confirmation Step. In this paper, we rely on $\mathbf{U}_A$ to verify that the observed gesture made by $\mathbf{U}_B$ indeed matches the expectation based on the H_{W_A}. This step could in future be automated by incorporating a gesture recognition system that would be able to track the precise location of $\mathbf{U}_B$'s hands. However, it is important to note that the current system also relies on the inherent human ability to detect anomalies and e.g. follow only the legitimate user, while detecting or ignoring any adversary's attempt to inject into the visual out-of-band channel. We thus leave this possibility for future work, together with the challenge of designing user gestures that would be particularly suitable for automated verification.

Designing for AR. While implementing the *HoloPair* prototype, we evaluated several design choices that had a large impact on the usability of the system. Besides the (incorrect) intuition that using a voice command to abort the protocol run would be more convenient, the largest usability improvement came as a the result of using only mixed-reality holograms to display information. Despite the initial expectation that important messages and objects would be best visible if always visible in a form of a Heads-Up-Display (HUD), reading information shown as real, mixed-reality objects with a fixed location in the environment proved to be significantly more natural. This is likely due to the fact that such holograms can be approached when needed, quickly and naturally glanced, or otherwise ignored.

Given that is one of the core new capabilities of the HoloLens device in comparison to previous AR devices that do not support precise *world anchoring*, we here emphasize the importance of re-evaluating existing design practices for

Table 1: Perfomance Impact of the HoloPair Prototype

Max δ GPU load	7% (from 22% idle)
Max δ CPU load	15% (from 36% idle)
Max RAM load	225 MB
Max δ Energy load	10% (from 50% idle)
Total network bandwidth	700 kB
Total application size	130 MB

various security primitives as they are being implemented in mixed reality.

9 RELATED WORK

General Device Pairing. Comprehensive overviews of a many different device pairing methods and their usability evaluations can be found in [15, 16]. For a recent work that extensively surveyed multiple shared secret confirmation steps for mobile devices and evaluated user's ability to detect potential attacks, see [24]. However, given that neither the two users is able to observe the output of both headsets, we are unable to directly apply previous work to the AR headset scenario. Secondly, in contrast to previous work in which users are required to e.g. copy some value from one device to the other, we make the assumption that the adversary can fully eavesdrop even the out-of-band channel. Considering the proliferation of augmented reality headsets and the fact that each of them has multiple front facing cameras, we believe this to be a necessary assumption.

Finally, despite previous proposals to e.g. shake mobile phones simultaneously and distribute the shared secrets this way [21], there is no obvious way to expose two AR headsets to the same outside conditions that a co-located adversary could not easily copy.

Given that even a simple comparison of visual outputs from two headsets is non-trivial, the problem of pairing AR headsets is likely to become an active topic of future security and usability research.

Securing augmented reality. The general topic of privacy and security of AR devices has recently gained traction, with researchers laying our general topics of interest [18] and discussing how to ensure that AR output on multi-app devices does not become dangerous [19]. Privacy research focused on building a hierarchy of visual recognizers to protect sensitive contexts [14] and, similarly, privately-preserving support for *3D browsers* [26], the ability to show applications *floating* at specific locations in a room (similar to Figure 1).. Moreover, a recent paper discussed the system and implementation-level security of the existing "AR browsers", platforms which allow overlaying 2D objects over a mobile device's camera output [22].

Secure pairing of AR headsets. Given the novelty of the AR headsets and the very recent availability of mixed reality headsets, the topic of AR pairing has not yet been extensively explored, with only a single related paper with the same focus [9]. However, the authors take a significantly different approach, by building their own hardware prototype which assumes that future AR headsets will have the support (multiple antennas) to perform precise wireless localization. To the best of our knowledge, this paper is the first security-focused research that uses the novel capabilities of Microsoft's HoloLens device to achieve usable and secure pairing of two AR headsets. Moreover, this work is the first proposes for a practically achievable AR headset pairing protocol that assumes only the existing capabilities of the HoloLens device.

10 CONCLUSION

In this paper we propose *HoloPair*, protocol and a system for secure and usable pairing of augmented reality headsets. We build a working prototype implementation of *HoloPair* using two Microsoft HoloLens headsets, full-fledged AR devices that have recently become available to developers in US and several other countries. By running a user study with a total of N=22 participants, we evaluate the feasibility of the proposed protocol in terms of security guarantees, usability, and prototype performance.

The experimental evaluation of the *HoloPair* prototype shows that participants with little or no prior experience using AR headsets are able to achieve high rates of detecting attack simulations or successfully pairing when no attack is simulated. Furthermore, the system is highly usable, as evident by short pairing times and high average scores achieved on the usability questionnaire, while having low computational requirements.

As shared collaboration that includes both holographic and real-world objects is at the core of current and future applications of AR, the ability for developers to secure direct connections between their users is of crucial importance.

Towards this goal, we have made our full prototype implementation and source code available online, and our prototype is started the process to have our solution integrated into the official HoloLens develoment kit. Considering the lack of practical proposals or implementations of secure device pairing, despite the rapid growth of AR platforms, we believe that this work is an important step in making future AR interactions secure and private from the start.

REFERENCES

[1] Gwen Ackerman and Dina Bass. 2017. Israeli Army Prepares Augmented Reality for Battlefield Duty. https://www.bloomberg.com/news/articles/2016-08-15/microsoft-s-hololens-technology-adopted-by-israeli-military. *Bloomberg Technology* (2017). Accessed: 2017-05-31.

[2] Imtiaj Ahmed, Yina Ye, Sourav Bhattacharya, N. Asokan, Giulio Jacucci, Petteri Nurmi, and Sasu Tarkoma. 2015. Checksum gestures: continuous gestures as an out-of-band channel for secure pairing. In *ACM International Joint Conference on Pervasive and Ubiquitous Computing*. 391–401.

[3] Aaron Bangor, Philip Kortum, and James Miller. 2009. Determining what individual SUS scores mean: Adding an adjective rating scale. *Journal of usability studies* 4, 3 (2009), 114–123.

[4] Alan Boyle. 2017. Microsoft HoloLens takes augmented reality to next level for warfighters. https://www.geekwire.com/2017/microsoft-hololens-takes-augmented-reality-next-level-americas-warfighters/. *GeekWire* (2017). Accessed: 2017-05-31.

[5] John Brooke. 1996. SUS: A quick and dirty usability scale. *Usability Evaluation in Industry* (1996).

[6] Nicholas Carlini, Pratyush Mishra, Tavish Vaidya, Yuankai Zhang, Micah Sherr, Clay Shields, David Wagner, and Wenchao Zhou. 2016. Hidden Voice Commands. In *25th USENIX Security Symposium (USENIX Security 16)*. USENIX Association, Austin, TX, 513–530.

[7] Jillian D'Onfro and Jay Yarow. 2014. Google Is Leading A 542 Million Investment In Magic Leap. http://www.businessinsider.com/magic-leap-google-investment-2014-10. *Business Insider* (2014).

[8] Michael Farb, Yue-Hsun Lin, Tiffany Hyun-Jin Kim, Jonathan M. McCune, and Adrian Perrig. 2013. SafeSlinger: easy-to-use and secure public-key exchange. In *The 19th Annual International Conference on Mobile Computing and Networking, MobiCom'13, Miami, FL, USA, September 30 - October 04, 2013*. 417–428.

[9] Ethan Gaebel, Ning Zhang, Wenjing Lou, and Y. Thomas Hou. 2016. Looks Good To Me: Authentication for Augmented Reality. In *Proceedings of the 6th International Workshop on Trustworthy Embedded Devices (TrustED '16)*. ACM, New York, NY, USA, 57–67.

[10] Lauren Goode. 2017. All Windows 10 PCs will support HoloLens apps next year. https://www.theverge.com/2016/8/16/12503868/microsoft-windows-holographic-windows-10-shell-features. (2017). The Verge online. Accessed: 2017-05-17.

[11] Google Inc. 2017. Tango Augmented Reality Computing Platform. https://get.google.com/tango/. *Online* (2017). Accessed: 2017-05-31.

[12] Microsoft Inc. 2017. HoloToolkit-Unity. https://github.com/Microsoft/HoloToolkit-Unity. *Online* (2017). Accessed: 2017-05-31.

[13] Unity Inc. 2017. Unity Game Development Platform. https://unity3d.com/. *Online* (2017). Accessed: 2017-05-31.

[14] Suman Jana, David Molnar, Alexander Moshchuk, Alan Dunn, Benjamin Livshits, Helen J. Wang, and Eyal Ofek. 2013. Enabling Fine-Grained Permissions for Augmented Reality Applications with Recognizers. In *22nd USENIX Security Symposium (USENIX Security 13)*. USENIX, Washington, D.C., 415–430.

[15] Ronald Kainda, Ivan Flechais, and A. W. Roscoe. 2009. Usability and Security of Out-of-band Channels in Secure Device Pairing Protocols. In *Proceedings of the 5th Symposium on Usable Privacy and Security (SOUPS '09)*. ACM, New York, NY, USA, Article 11, 12 pages. https://doi.org/10.1145/1572532.1572547

[16] Alfred Kobsa, Rahim Sonawalla, Gene Tsudik, Ersin Uzun, and Yang Wang. 2009. Serial Hook-ups: A Comparative Usability Study of Secure Device Pairing Methods. In *Proceedings of the 5th Symposium on Usable Privacy and Security (SOUPS '09)*. 10:1–10:12.

[17] Elizabeth Landau. 2017. 'Mixed Reality' Technology Brings Mars to Earth. https://www.jpl.nasa.gov/news/news.php?feature=6220. *NASA Jet Propulsion Laboratory* (2017). Accessed: 2017-05-31.

[18] Kiron Lebeck, Tadayoshi Kohno, and Franziska Roesner. 2016. How to safely augment reality: Challenges and Directions. In *Proceedings of the 17th International Workshop on Mobile Computing Systems and Applications*. ACM, 45–50.

[19] Kiron Lebeck, Kimberly Ruth, Tadayoshi Kohno, and Franziska Roesner. Securing Augmented Reality Output. In *Proceedings of the 38th IEEE Symposium on Security and Privacy (Oakland 2017)*.

[20] David Lumb. 2017. Microsoft HoloLens becomes an AR assistant for spinal surgery. https://www.engadget.com/2017/05/05/microsoft-hololens-becomes-an-ar-assistant-for-spinal-surgery/. *Engadget* (2017). Accessed: 2017-05-31.

[21] R. Mayrhofer and H. Gellersen. 2009. Shake Well Before Use: Intuitive and Secure Pairing of Mobile Devices. *IEEE Transactions on Mobile Computing* 8, 6, 792–806. https://doi.org/10.1109/TMC.2009.51

[22] Richard McPherson, Suman Jana, and Vitaly Shmatikov. 2015. No Escape From Reality: Security and Privacy of Augmented Reality Browsers. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. International World Wide Web Conferences Steering Committee, 743–753.

[23] Mariella Moon. 2017. Apple has NASA minds working on its AR glasses. https://www.engadget.com/2017/04/25/apple-augmented-reality-glasses-nasa-jeff-norris/. (2017).

[24] Joshua Tan, Lujo Bauer, Joseph Bonneau, Lorrie Faith Cranor, Jeremy Thomas, and Blase Ur. 2017. Can Unicorns Help Users Compare Crypto Key Fingerprints?. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. 3787–3798.

[25] Serge Vaudenay. 2005. *Secure Communications over Insecure Channels Based on Short Authenticated Strings*. Springer Berlin Heidelberg, Berlin, Heidelberg, 309–326.

[26] John Vilk, David Molnar, Benjamin Livshits, Eyal Ofek, Chris Rossbach, Alexander Moshchuk, Helen J. Wang, and Ran Gal. 2015. SurroundWeb: Mitigating Privacy Concerns in a 3D Web Browser. In *Proceedings of the 2015 IEEE Symposium on Security and Privacy (SP '15)*. IEEE Computer Society, Washington, DC, USA, 431–446.

[27] Tom Warren. 2017. Microsoft reveals Dell and Asus Windows Mixed Reality headsets. https://www.theverge.com/2017/5/31/15717478/dell-asus-windows-mixed-reality-headsets-features. *The Verge* (2017).

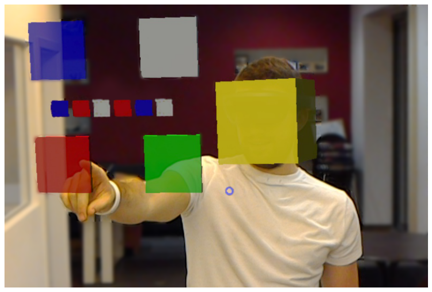
APPENDIX

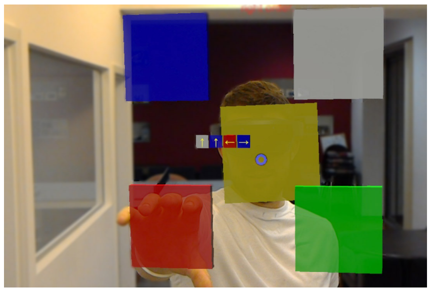
A SUS QUESTIONS

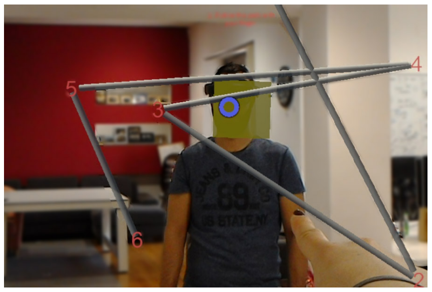
The System Usability Scale [5] is a widely used 10 statement questionnaire, with Likert-scale answers (0 - "Strongly disagree" to 4 - "Strongly agree"). The overall usability score for an individual user is computed by summing the answers to odd-positioned questions (Q1, Q3, ...) and subtracting the answers on even-positioned questions (Q2, Q4, ...). The score is finally centered and scaled to [0, 100] by adding 20 and multiplying the resulting value with 2.5.

The 10 SUS questions are:

(1) I think that I would like to use this system frequently.
(2) I found the system unnecessarily complex.
(3) I thought the system was easy to use.
(4) I think that I would need the support of a technical person to be able to use this system.
(5) I found the various functions in this system were well integrated.
(6) I thought there was too much inconsistency in this system.
(7) I would imagine that most people would learn to use this system very quickly.
(8) I found the system very cumbersome to use.
(9) I felt very confident using the system.
(10) I needed to learn a lot of things before I could get going with this system.


(a) *"Touch the larger cubes according to the color sequence of length N."*


(b) *"Touch the cube and then swipe in one of 4 directions as indicated by the arrows."*



(c) *Follow the shape with your hand."*

Figure 9: Three different versions of the shared secret confirmation step that we use in this paper. In (a), $\mathbf{U}_A$ confirms that $\mathbf{U}_B$ is touching the correct sequence of colored cubes. In (b), after touching any of the cubes, $\mathbf{U}_B$ is required to also swipe their hand in one of the 4 directions (depicted by arrows). Finally, in (c), $\mathbf{U}_B$ follows the 3D shape in front of him with his hand, while $\mathbf{U}_A$ confirms that the shape shown on his $\mathbf{D}_A$ matches the hand movement (and correspondingly the shape on $\mathbf{D}_B$).

B ALTERNATIVE SHARED SECRET CONFIRMATION STEPS

HoloPair does not rely on any specific form of the shared secret confirmation step. Consequently, in our research we designed, developed, and evaluated three different versions of this confirmation step, shown in Figure 9. We now describe the initial two schemes that were tested, before deciding to focuse on the final scheme (based on *virtual pipes*), that was described throughout the paper.

(a) - "Cubes": Our initial design choice was based on the idea of having a shared keyboard that one user would touch, while the other observes the gestures and verifies that they are correct. As shown in Figure 9a, in order to ensure a simple user experience, $\mathbf{U}_B$ is required to look at the sequence of colors shown at the bottom of their screen and accordingly touch one of the four larger cubes to communicate the value of the weak-hash generated on his headset. At the same time, $\mathbf{U}_A$'s device independently generates the sequence of N and shows the four larger cubes at the same location as $\mathbf{U}_B$'s device. $\mathbf{U}_A$ observes if $\mathbf{U}_B$'s sequence of colors indeed match, and thus verifies high probability that the exchanged public keys are indeed authentic.

This results in the total number of different configurations that can be presented with N colors to be 4^N, and correspondingly, the expected probability of successfully guessing an alternative pair of public keys for an attacker to be:

$$P_a(N) = \frac{1}{4^N}$$

(b) - "Cubes with Arrows": In an extension of the first variant, shown in Figure 9b, the weak-hash that is independently generated on both devices also specifies one of 4 directions for each of the colors in the sequence. Correspondingly, $\mathbf{U}_B$ is required not only to touch the larger cube of the same color, but also to make a hand movement in the direction indicated on his sequence. The reasoning behind this version is that additional hand movement does not significantly impact the usability of the secret confirmation step, while it, at the same time, squares the number of possible weak hashes that users can communicate using a sequence of length N.

As $\mathbf{U}_B$ is additionally required to move their hand in one of the four directions, this further increases the number entropy of the weak-hash to $(4 \times 4)^N$ and decreases the probability of a successful attack to:

$$P_b(N) = \frac{1}{(4 \times 4)^N} = \frac{1}{16^N}$$