Looks Good to Me (LGTM): Authentication for Augmented Reality Using Wireless Localization and Facial Recognition

Ethan D. Gaebel

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science and Applications

Wenjing Lou, Committee Chair
Ing-Ray Chen
Guoqiang Yu

May 15, 2016
Falls Church, Virginia

Keywords:

Looks Good to Me (LGTM): Authentication for Augmented Reality Using Wireless Localization and Facial Recognition

Ethan D. Gaebel

(ABSTRACT)

Augmented reality is a powerful new computer interface paradigm that is going to reshape how people interact with computers. The diverse hardware on typical augmented reality headsets presents an opportunity to the security community to re-make how authentication is carried out, making it more usable for the everyday user and moving authentication away from the cloud and back to face-tof-face interactions (when users are face to face anyway). In this paper we explore an authentication protocol to be used when users are face to face and have augmented reality headsets. This protocol uses the same idea behind how humans identify speakers: localizing a wave and comparing its location to a face's location, except here we deal with radio waves instead of sound waves. This protocol is designed to minimize the work required by the user, thus increasing the likelihood that it will be used and decreasing the likelihood of user error. We provide an implementation of the protocol as well.

Security-based abstract and stuff!!!

This work was supported in part by the National Science Foundation under Grants CNS-1443889, CNS-1405747, CNS-1446478, and CNS-1343222.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Augmented reality is one of the most exciting new technologies on the horizon, promising three dimensional interfaces that one or more users can directly interact with. Augmented reality is most commonly provided via a head mounted display with translucent lenses having digital content in two or three dimensions rendered on top of the real world. Many of these head mounted displays will be full stand-alone computers equipped with powerful processors, wireless communications, one or more high resolution cameras, and depth sensors. Stand alone head mounted displays require much of this sophisticated technology to provide fully interactive augmented reality which, at a minimum, requires: hand tracking, graphics processing, and wireless communication, since connectivity is a requirement for any consumer device today.

Users of augmented reality headsets have the same need that users of any other consumer electronic device have today: the need to send content to others. In the case of augmented reality this content will often be far richer, and thus larger, than we've seen on other platforms, since users will be creating, viewing, and working with three dimensional objects, which inherently have more information associated with them than their equivalent two dimensional representations. It is also likely that users will want to share content face-to-face, providing 3D objects that both users can see and/or interact with in real time; most smart phone users have looked up a video or picture to show someone before. In fact, we can expect this type of sharing to increase with augmented reality so that users are sharing app interfaces, sharing group photos in real-time, and passing secret three dimensional jokes back and forth constantly just like they send snaps on Snapchat today.

Content sharing between users in close proximity is an interesting, and well studied [$CITATIONS$], problem. Typical content sharing schemes today take advantage of pre-existing infrastructure via cell towers, wireless access points, and the Internet. This makes very good sense considering the most common use-case for digitally sharing content is to share it with those who aren't present; you wouldn't send someone standing next to you a video. But with augmented reality you would send someone sitting next to you a video so that the two of

you can watch it together through your respective headsets. Since this type of sharing will be so pronounced I think it deserves separate consideration from the general content sharing problem, which is why I think that point-to-point wireless communication should be used for such content sharing, particularly when the content is stored locally on a user's device, as would be the case with photographs, three-dimensional scene captures, videos, and user created three dimensional content. By sharing content directly from person-to-person network resources are saved, money may be saved if charge-by-data-usage plans are being used for Internet access. Person-to-person content sharing is robust in the face of local or global infrastructure failures, and content sharing privacy details, such as who is communicating with who, what is being communicated, and when the communication is occurring, are kept between the two users doing the sharing instead of being shared with whatever entity (or entities) is (are) providing the service.

There are many options for point-to-point wireless communication but a full discussion and evaluation of these options is outside the purview of this thesis. However it is relevant to briefly mention some major enabling technologies so as to convince the reader of the feasibility of such a design, so I will briefly mention a few common technologies. WiFi Direct [$CITATION$] is perhaps the simplest since any WiFi-capable device can take advantage of it with the right software. WiFi Direct also has direct API support in the popular Android operating system. Bluetooth also includes a point-to-point protocol, with support in Android, iOS, and most modern computers.

When working in a point-to-point wireless communications scenario authentication can be tricky. It will often be the case that two users have not used their devices to communicate before, so there will be no pre-existing security context (such as a pin, key, token, etc) between the two devices.

Device pairing is the area dedicated to authenticating devices without prior security context, and there have been many schemes introduced to address this problem on devices with myriad hardware features and constraints. To give a quick idea of the breadth of the techniques proposed, some schemes have involved: screens with changing bar codes, infrared communication, blinking lights, communicating cryptographic keys using speakers and microphones, and numerical pin entry. All of these methods rely on communication over one or more out-of-band channels, which is a channel using human sensory capabilities to authenticate other communication channels which are imperceptible to humans, which is usually the wireless channel. To authenticate the human imperceptible channel some key is transmitted over the out-of-band channel which is then used for authentication. The simplest, and most common, example is that of a numerical pin. A pin may be permanently bound to a device without a user interface and the user enters the pin into another device with a user interface to perform authentication; this is the case with Bluetooth headsets.

These out-of-band channels explicitly require a human to participate, and it may be expected that the human actor has some impact on the security of the scheme as a result. This intuition has been validated in [$CITATIONS$] where comparative user studies

were conducted between several different device pairing techniques. In [$CITATION --UsabilityAnalysisofSecureDevicePairingMethods$] two rounds of usability tests were performed for two different methods: compare-and-confirm and select-and-confirm, where users had to compare a randomly generated pin presented on each device for equality and confirm if they matched and select a pin from a list on one device sent to it from the first device and confirm if they matched, respectively. Each method was measured by fatal error rate and total user error rate, where a fatal error is an error that results in a security breach and total user error is all the fatal errors combined with all errors that do not result in a security breach (only user annoyance). The two rounds of usability tests differed in the user interface the study participants used, while also being performed on different users. In the first round compare-and-confirm had a fatal error rate of 20% and select-and-confirm had a fatal error rate of 12.5%, while compare-and-confirm had a 20% total error rate and select-and-confirm had a 20% total error rate. The researchers suspected that the user interface could be the problem and so improved the interfaces as well as lengthening the pins from 4 numbers to 6 numbers, the idea being that a longer pin will force the user to focus more on the task making them less likely to make a mistake. In the second round compare-and-confirm had a fatal error rate of 0% and select-and-confirm had a fatal error rate of 5%, while compare-and-confirm had a 2.5% total error rate and select-and-confirm had a 7.5% total error rate. So we see that these results mirror our intuition regarding usability's effect on the security of device pairing schemes. Furthermore, making authentication easier for the user is a worthy goal in-of-itself. A hallmark of technology in general has been making certain things easier for the masses, so it should be with security technology.

Augmented reality presents a new computing platform with many hardware features as a requirement that have not been previously seen on consumer devices. These hardware features present an opportunity for a better device pairing scheme that is both more secure and more usable. Think of how two people who have never met before authenticate one another while speaking. Sound waves are localized and paired with the face we see at the point of localization. Afterwards we know the sound of the person's voice and can recognize it without seeing them. Can we do something similar with the hardware available in an augmented reality headset? Yes. We can localize the wireless signal and digitally overlay its location with a box or other type of indicator on reality for the user to view, and use facial recognition to locate a face fitting a model transmitted to our device and overlay a box around the face recognized in the current frame. The user can then verify on the digital overlay that the wireless signal is emanating from the person we would like to communicate with. If the user finds that the person our device recognizes is who they want to communicate with and that the wireless signal is indeed coming from them, the user can say to themselves "looks good to me!" and indicate that the other user has been authenticated. This is the basic view of how my system will work from the user's perspective, but lets move on to the next section to formalize this protocol and everything that happens under the hood.

Chapter 2

Protocol Design

2.1 System Model

First we must adequately understand the scenarios where LGTM is applicable. This requires us to go down the list of: hardware requirements, security context, the users' context, and any other assumptions made. LGTM is specifically designed for authenticating two users, Alice and Bob.

LGTM assumes that users are equipped with head mounted augmented reality rigs each of which posses: wireless communications hardware with support for a point-to-point communication protocol, a high definition video camera, and a translucent display directly in front of the users' eyes that is capable of displaying digital objects on top of the physical world (this last requirement is satisfied simply by the users possessing head mounted augmented reality rigs, but we want to be thorough).

The two hardware devices are assumed to have no prior security context, which means the two devices do not possess any information which can be used to prove the authenticity of their claimed identity to the other. A pre-shared key is a common example of a piece of information which can be used as preexisting security context.

An important piece of information that each headset is assumed to possess is a facial recognition model capable of recognizing the user of the headset. That is, Alice's headset has a facial recognition model which can recognize her and Bob's headset has a facial recognition model which can recognize him. Training for these models can be done easily in a mirror.

As for the users themselves, Alice and Bob are assumed to be in sight of each other and would like to share some sort of digital content with one another such as: a shared application experience, a shared video, shared music, shared holograms, or basic messaging. No assumptions are made of Alice and Bob's relationship with each other: they may be friends, acquaintances or strangers.

No assumption regarding access to the Internet is made.

2.2 Security Objectives

The primary security objective is for Alice and Bob to correctly authenticate one another's wireless signals so that they can engage in secure point-to-point wireless communication. Recall from the system model section that Alice and Bob do not share any security context, so this authentication must be bootstrapped.

A secondary objective is for Alice and Bob to correctly select which (user, wireless signal) pair they want to communicate with. This objective is directly related to the first but we feel it warrants distinction as a worthy problem in its own merit since there are many practical attacks which rely upon tricking users into selecting the wrong thing [$CITATIONS, DIVERSEONES$]. We refer to this as the selection problem.

There is a final security objective which is perhaps more accurately described as a privacy objective. That objective is for there to be a guarantee that no outside parties are aware of the communication or content patterns present for any user of LGTM. Specifically, this means that it is not possible for an outside entity, one not directly engaging with Alice, to determine who she communicates with or what she communicates.

2.3 Threat Model

Attackers have quite a bit of power when dealing with a wireless channel since the communications medium is both public and localized. Attackers have the capability to: jam communications, perform denial of service attacks by flooding the channel with packets constructed to fit the protocol, eavesdrop on all packets transmitted, and replay packets collected from eavesdropping in any order. Further, the attacker may have multiple transceivers and their disposal, so attacks can be coordinated between more than one node. Attackers may use these capabilities to accomplish one or several goals. An attacker may want to impersonate Alice to send Bob falsified content. An attacker may want to impersonate both Alice and Bob to perform a man-in-the-middle attack allowing them to eavesdrop on encrypted communications between Alice and Bob. The attacker may simply want to prevent communication between Alice and Bob. Or the attacker may want to do any of the above steps with the higher goal of exploiting some subsystem.

2.4 Protocol

Consider Alice and Bob to be person-device tuples, that is, Alice and Bob are people wearing augmented reality head-mounted displays equipped with wireless communications, high definition cameras, and translucent displays very close to their eyes. Alice and Bob meet in person and would like to establish communication, but they have never communicated on these devices, so they have no pre-existing security context. Alice and Bob both initiate the pairing protocol on their respective devices via a button which may be physical or digitally rendered. Alice and Bob generate new public key-private key pairs and each broadcast their public keys, unencrypted, into the wireless channel. Bob encrypts a facial recognition model to recognize his own face with Alice's public key and sends it. Alice decrypts Bob's message to retrieve his facial recognition parameters and also saves the channel state information retrieved from her wireless card for the packets holding Bob's facial recognition parameters. Alice encrypts facial recognition parameters for her own face using Bob's public key and sends the result. Bob receives Alice's encrypted facial recognition parameters and decrypts them to retrieve them while also saving the channel state information collected from Bob's wireless card for the packets holding Alice's facial recognition parameters. Alice computes Bob's location using the channel state information and displays the location on Alice's translucent display in the form of a circle (to include some error tolerance). Alice also runs a facial recognition algorithm using Bob's transmitted facial recognition parameters to identify any faces which match Bob's. If any faces are recognized a a box is displayed around the face on Alice's translucent display. It is checked if there is an overlap between the box around Bob's face and the circle around Bob's computed signal transmission location. If there is an overlap between the two areas then Alice is prompted to confirm whether the correct person has been identified. Bob performs the same steps, computing the location of Alice's signal origin, running a facial recognition algorithm using Alice's transmitted parameters, checking for overlap of the two, and being prompted to confirm that the intended recipient, Alice, has been selected.

Alice and Bob have now authenticated each others' wireless signals and can perform some key exchange protocol (like Diffie-Hellman) to begin encrypted communication.

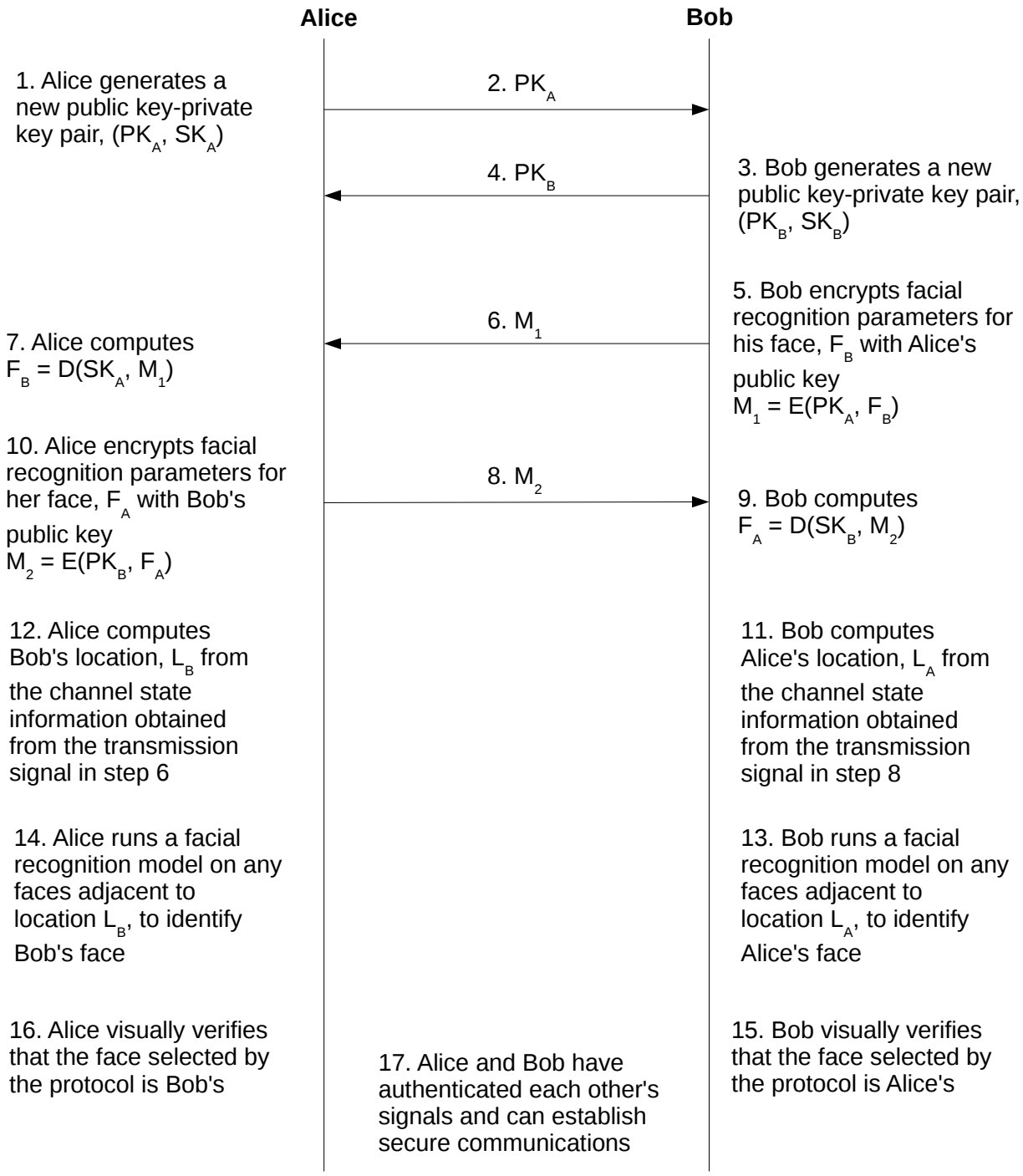

Figure 2.1: Looks good to me protocol diagram...

2.5 Analysis

2.5.1 Security

2.5.2 Usability

2.5.3 Privacy

2.5.4 Potential Alterations

Chapter 3

Implementation

Chapter 4

Experiments

Chapter 5

Conclusion