KBID: Kerberos Bracelet Identification (Short Paper)

Joseph Carrigan, Paul Martin, and Michael Rushanan

Johns Hopkins University {joseph.carrigan,pmartin,mrushan1}@jhu.edu

Abstract The most common method for a user to gain access to a system, service, or resource is to provide a secret, often a password, that verifies her identity and thus authenticates her. Password-based authentication is considered strong only when the password meets certain length and complexity requirements, or when it is combined with other methods in multi-factor authentication. Unfortunately, many authentication systems do not enforce strong passwords due to a number of limitations; for example, the time taken to enter complex passwords. We present an authentication system that addresses these limitations by prompting a user for credentials once and then storing an authentication ticket in a wearable device that we call *Kerberos Bracelet Identification* (KBID).

Keywords: authentication, kerberos, wearables, passwords

1 Introduction

The use of modern computer systems almost always requires that a user prove their identity through some process of authentication. Specifically, a user can authenticate using methods such as public key authentication, biometrics, and passwords; something you have, are, or know, respectively. Password-based authentication remains the most widely used option for authentication because of its ease of use and simple design.

However, passwords have a human factor weakness as users often choose passwords that are too simplistic and easily guessed [9]. Administrators and systems require users to select more complex passwords as a consequence; thus, decreasing user satisfaction as password selection becomes seemingly difficult. Worst yet, complex passwords interfere in critical workflow such as clinical care where a patient's need is most urgent.

In this paper we describe an authentication system that requires the user to enter a password as infrequently as once a day. Specifically, authentication information is stored on a wearable device, a bracelet in our case, and is transmitted to devices to which the user wishes to authenticate. The transmission between the bracelet and device is achieved via using the user's body as a communication medium [1,2]. Our goal is to reduce the impact on user satisfaction and workflow by removing most of the difficulty of using a complex password.

While we focus on authentication in the medical community use case, we anticipate that other areas such as the financial sector may benefit from our system. In addition, it is important to note that this system is not a two-factor authentication solution. The bracelet is not a biometric component and does not provide any additional information outside of what it stores. It is meant to enhance the user experience and encourage the use of complex passwords. Finally, KBID is a work in progress.

2 Background

KBID originates from the idea of integrating a wearable device to achieve some additional property in an authentication system (e.g., de-authentication). In particular, we are inspired by the design of zero-effort bilateral recurring authentication (ZEBRA) by Mare et al. [7]. In ZEBRA, a user wears a bracelet that encapsulates a wireless radio, accelerometer, and gyroscope; these components record and transmit wrist movements to a computer system that is currently being used. The computer system continually compares received movement measurements to input it receives from it's keyboard and mouse. If these two measurements are not correlated, the current session is de-authenticated.

At the time, ZEBRA was only envisioned as a method to de-authenticate a user from a computer system and did not include a way to initially authenticate the user to the system. Assuming that the user has already accepted wearing a device that will effectively de-authenticate them, adding functionality to rapidly authenticate them only increases the usefulness of the system.

We avoided using radio frequency (RF) emissions for two reasons. First, RF by it's nature emits information into the environment. That information, once emitted, can be received by various means. Second, RF relies on the underlying communication being secure. If a security flaw is discovered in an RF communication framework, e.g. Bluetooth low energy (BLE) [8], then the systems as it exists could be vulnerable to the flaw. Specifically, there is no need to alter or even monitor the information that is exchanged between the computer system and a wearable device. An attacker would only need to extend the range of the wireless communication in order to gain access to the computer system.

We instead use body-coupled communication (BCC), or transmission of information over the human body as a medium. We are not the first to use BCC to transmit a secret. For example, Chang et al. introduce a system for key exchange over a body area network [2]. By applying a very small voltage to the tissue of a dead mouse, they were able to communicate at a rate of 5Hz or 5 bits per second. However, this data rate is not acceptable for our work as we would need to communicate authentication data of at least 256 bits, and this would take nearly a minute to transmit.

We designed our authenticated bracelet to be non-transferrable (i.e., authenticating and then giving the bracelet to someone else). To support this feature, we zero all authentication information upon bracelet removal.

3 Related Work

In addition to the ZEBRA, which we have described previously, there have been several previous attempts at developing wearable-authentication technology. Two of note include the Bionym Nymi [5] and the Intel Authentication Bracelet [6]. The Bionym Nymi is an authentication wristband that broadcasts a digitally signed authentication signal derived from a user's heartbeat to nearby devices using BLE [4]. The Intel Authentication Bracelet requires a user to log in to a system with a standard password. A credential is then transmitted to the bracelet using BLE. This credential is then broadcast to nearby bracelet-enabled devices in order to allow password-less login.

Limitations of Existing Work Existing authentication wearables use wireless communication technology (typically BLE) to broadcast their authentication credentials. Thus the devices are likely vulnerable to ghost-and-leech attacks [3]. Ghost-and-leech attacks occur when an attacker uses a more powerful radio transmitter than the transmitter found on a wireless device in order to capture and rebroadcast the wireless signal in order to fool a target into believing that the wireless device is in closer proximity to the target than it actually is.

4 Threat Model

As a hardware and software solution, the threat model for KBID includes many subjects that apply to any such system. These include attack types (denial of service, message forging or tampering, hardware tampering, and others) as well as a study of potential adversaries and other topics. Here we focus on two threats that are unique to KBID. These threats require an active adversary that has the ability to get close to where KBID is used. It is possible to impersonate an authentication module. An attacker could use a counterfeit authentication module that issues *Get Status* commands when a user touches something connected to it, e.g. a door knob. We discuss mitigating this vector below. Second, while we are using body coupled communication to transmit data without emitting RF, it could be the case that the user's body acts as a broadcast antenna and emits the data into the environment where it could conceivably be intercepted. We plan on assessing the reality of this treat during the development of the next prototype.

5 Design

Here we describe in detail the design and implementation of the KBID system. First we discuss the high level design where we explain the four major components of the system. Next, we discuss the interface designs and the communication protocols between the major components. Finally, we discuss the system workflow.

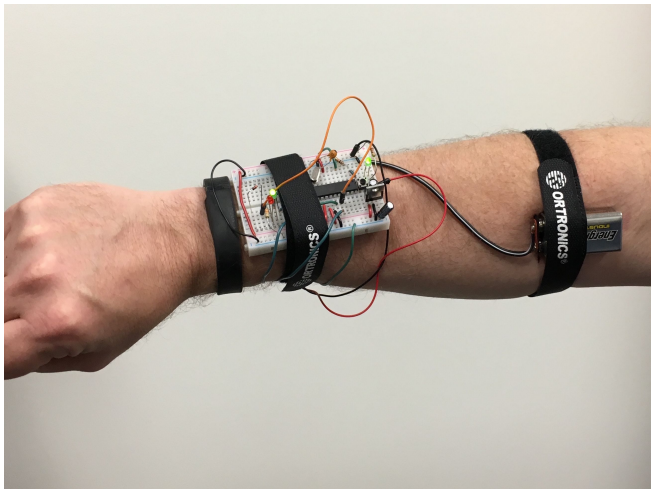

Figure 1. KBID Prototype Bracelet

5.1 High Level Design

The system is composed of four main parts: a bracelet (Figure 1), an authentication module (Figure 2) an authentication client, and a Kerberos authentication server. The bracelet is a wearable device that fastened to the user's wrist. The bracelet makes contact with the user's skin and applies a signal directly to the user's skin. The authentication module has a sensor with a button under it. When the user touches the sensor and depresses the button, the authentication module initiates communication with the bracelet. The authentication module is attached via RS-232 serial to the computer system to which the user wants to authenticate. A workstation hosts the authentication client. The client monitors the serial connection for data and when necessary, opens a connection to the Kerberos server for authentication. Finally, the Kerberos server is a default installation and uses the default implementation of the authentication protocol.

5.2 Interfaces and Communication

The KBID system includes three interfaces. The interface between the bracelet and the authentication module takes place over the user's skin. The interface between the authentication module and the authentication client takes place over RS-232 serial. Finally, the interface between the authentication client and the Kerberos server uses the network. Since the communication between a client and a Kerberos server is well documented, we will not discuss it in this paper.

Bracelet to Authentication Module The communication protocol between the bracelet and the authentication module is a very lightweight protocol. The messages that the bracelet sends to the authentication modules are called *statuses*.

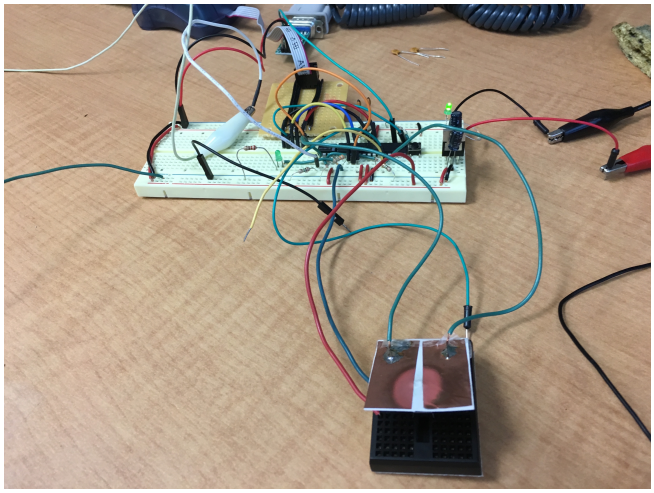

Figure 2. KBID Prototype Authentication Module

The messages that the authentication module sends to the bracelet are called *commands*. Each message sent over this interface is a length delimited series of bytes. A status message had the following structure: [Status ID] [Device ID] [Data Size (in bytes)] [Data]. The bracelet will send one of two statuses, *authenticated* or *un-authenticated*. If the status is authenticated, the authentication data will be transmitted in the data field.

A command message has the following structure: [Command ID] [Device ID] [Payload Size (in bytes)] [Payload]. The authentication module will send three commands: *Get Status*, *Set Token*, and *De-authenticate*. A Get Status command causes the bracelet to respond with a status message. A Set Token command causes the bracelet to store the payload in memory as authentication data and set its status to authenticated. A De-authenticate command causes the bracelet to clear any token it has and set its status to un-authenticated.

Authentication Module to Authentication Client The authentication module and the authentication client communicate status and command messages as well. The authentication module can send three statuses to the authentication client. First is the *Un-authenticated Bracelet* message. This message is sent to the authentication client when the authentication module receives an un-authenticated status from a bracelet.

Next the authentication module can send an *Authenticated Bracelet* status to the authentication client. It will send this status when the bracelet sends a status of authenticated. The Authenticated Bracelet status will contain the ticket information that was in the token section of the bracelet's message.

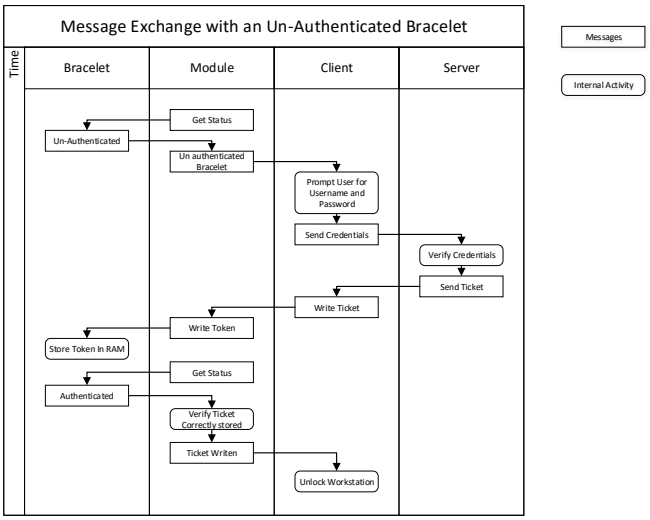

Figure 3. Un-Authenticated Message Exchange

Finally, the authentication module can send a *Ticket Written* status to the authentication client. This is a message that lets the client know that the ticket information has been successfully written to the bracelet.

The authentication client sends two commands to the authentication module. First the *Write Ticket* command. This instructs the authentication module to pass the ticket included in the command to the bracelet with a Set Token command. The authentication client can also send a *De-authenticate Bracelet* command. This instructs the authentication module to issue a De-Authenticate command to the bracelet.

5.3 System Workflow

The system workflow can be described in two use cases. For the sake of brevity we do not include any error handling. In the first use case (Figure 3) the user is wearing a bracelet but the bracelet is not yet authenticated. The users touches the sensor on the authentication module, the authentication module sees that the user's bracelet is not authenticated and relays this information to the authentication client. The client prompts the user for their username and password. The client verifies this information with the Kerberos server, then instructs the user to touch the sensor on the authentication module again. The client then instructs the authentication module to write the ticket to the bracelet. Once the ticket has been written, the client unlocks the workstation.

In the second use case (Figure 4) the user has an authenticated bracelet. The user touches the sensor on the authentication module. The module asks for a status, and the bracelet provides it with the token it has stored. The module passes this information along to the client which interprets the token

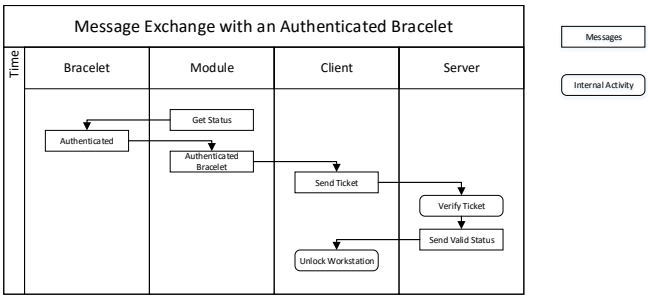

Figure 4. Authenticated Message Exchange

as a Kerberos ticket. The client verifies the ticket with the Kerberos server and unlocks the workstation. Our goal is to perform this use case in less than one second.

6 Experiments and Results

6.1 Prototype

The hardware prototypes for the bracelet and the authentication module are based on the Atmel ATMega328 microcontroller operating at 20 MHz and an LM358AN Amplifier. Both the bracelet and the authentication module have copper pads that make contact with the user's skin. The signal from the skin is fed into the amplifier and the signal to the skin is driven by setting a pin on the microcontroller. We also built a resistive analogue to represent the resistance from a user's wrist to their finger tip. The prototype for the authentication client has been written in Python.

6.2 Results

Initial results are encouraging. We are able to send commands from the authentication module to the bracelet. The bracelet can correctly interpret those commands and it responds when issued a Get Status command. The time elapsed for a Get Status command and a status message with a 256 byte token is approximately 500 milliseconds. We also implemented the functionality that clears the authentication information when the bracelet is removed. This was accomplished by using one of the hardware interrupts on the microcontroller.

6.3 Hurdles

We encountered two major hurdles during the development of the first prototype. First, in order to successfully send a signal the bracelet and the authentication module must have a common reference for voltage. Second, while we have been

able to get the signal to transmit from the authentication module to the bracelet, we have not been able to get the signal to travel in the opposite direction and arrive in a way that the signal can be interpreted by the authentication module. To solve these issues, we plan to either improve the performance of the amplifier or change the method by which the signal is transmitted over the user's skin to capacitive coupling.

7 Future Work

Future work will focus on improving the system. We plan to implement the system using a microcontroller that can store larger keys. The Atmel ATMega328 only has 2 kilobytes of RAM. Since the systems needs some RAM to perform operations, the key that is stored in RAM is limited in size to about 1 kilobyte. This is not sufficient space to store authentication information in real world environments. We also plan on hardening the system by adding pre-shared message authentication codes (MACs) to protect against replay and device impersonation attacks.

8 References

References

1. Barth, A.T., Hanson, M.A., Powell, H.C., Unluer, D., Wilson, S.G., Lach, J.: Body-coupled communication for body sensor networks. In: Proceedings of the ICST 3rd International Conference on Body Area Networks (2008), http://dl.acm.org/citation.cfm?id=1460257.1460273
2. Chang, S., Hu, Y., Anderson, H., Fu, T., Huang, E.Y.L.: Body area network security: Robust key establishment using human body channel. In: Proc. 3rd USENIX Workshop on Health Security and Privacy (HealthSec) (Aug 2013), https://www.usenix.org/conference/healthsec12/workshop-program/presentation/Chang
3. Czeskis, A., Koscher, K., Smith, J.R., Kohno, T.: Rfids and secret handshakes: defending against ghost-and-leech attacks and unauthorized reads with context-aware communications. In: Proceedings of the 15th ACM conference on Computer and communications security. pp. 479–490. ACM (2008)
4. Gomez, C., Oller, J., Paradells, J.: Overview and evaluation of bluetooth low energy: An emerging low-power wireless technology. Sensors 12(9), 11734 (2012), http://www.mdpi.com/1424-8220/12/9/11734
5. Goode, A.: Bring your own finger–how mobile is bringing biometrics to consumers. Biometric Technology Today 2014(5), 5–9 (2014)
6. Krzanich, B.: Intel developer forum san francisco opening keynote. Tech. rep., Intel Corporation (2015)
7. Mare, S., Markham, A., Cornelius, C., Peterson, R., Kotz, D.: Zebra: Zero-effort bilateral recurring authentication. In: Security and Privacy (SP), 2014 IEEE Symposium on (May 2014)
8. Ryan, M.: Bluetooth: With low energy comes low security. In: Proceedings of the 7th USENIX Conference on Offensive Technologies. USENIX Association (2013), http://dl.acm.org/citation.cfm?id=2534748.2534754

9. Yan, J., Blackwell, A., Anderson, R., Grant, A.: Password memorability and security: Empirical results. IEEE Security and Privacy (Sep 2004), http://dx.doi.org/10.1109/MSP.2004.81