

# BAD: Autenticación mediante la utilización de señales EMG

Bezdjian Alejandro (Leg. N° 52108)      Marzoratti, Luis (Leg. N° 54449)

Julio 2018

# Índice

<b>1. Abstract</b>	<b>4</b>
<b>2. Introducción</b>	<b>4</b>
<b>3. Justificación del trabajo</b>	<b>5</b>
<b>4. Estado del arte</b>	<b>6</b>
4.1. Electromiografía (EMG) . . . . .	6
4.2. Electroencefalografía (EEG) . . . . .	6
4.3. Sistema de Autenticación por Biometría . . . . .	7
<b>5. Dispositivos para Captura de Señales Biométricas</b>	<b>8</b>
5.1. TrueSense . . . . .	8
5.1.1. Problemas encontrados . . . . .	9
5.2. Olimex EMG/EEG Shield . . . . .	12
5.3. Muse headband . . . . .	13
5.4. Conclusión . . . . .	14
<b>6. Análisis de la señal EMG</b>	<b>14</b>
6.1. Comparación de las señales . . . . .	16
<b>7. BAD API</b>	<b>17</b>
7.1. Arquitectura . . . . .	18
7.2. Comunicación . . . . .	18
7.3. MQTT . . . . .	19
7.3.1. Canal “Status” . . . . .	20
7.3.2. Canal “Signal” . . . . .	20
7.4. Comunicación entre los usuarios, clientes y la BAD API . . . . .	20
7.5. BAD-DEVICE-I: Prototipo del dispositivo . . . . .	21
7.6. Integración por parte de los clientes y kickstart . . . . .	22
7.7. Interfaz BAD API . . . . .	24
7.7.1. GET /api/v1/signals/:uuid . . . . .	24
7.7.2. POST /api/v1/signals/compare . . . . .	24
7.7.3. GET /api/v1/devices . . . . .	25
7.8. Interfaz BAD-DEVICE-I . . . . .	25
7.8.1. POST /api/v1/start . . . . .	25
7.8.2. POST /api/v1/stop . . . . .	26
7.8.3. POST /api/v1/cancel . . . . .	26
7.8.4. GET /api/v1/read . . . . .	26
<b>8. Threat model</b>	<b>27</b>
8.1. Comunicación entre el dispositivo y la computadora del usuario . . . . .	27
8.2. Entre el servidor del cliente y la BAD API . . . . .	28
8.3. Entre el dispositivo y la BAD API . . . . .	28
8.4. Entre la computadora del usuario y el servidor del cliente. . . . .	29
8.5. Guardado del ID de la señal por parte del cliente . . . . .	29
8.6. Guardado de las claves y la señal en el dispositivo . . . . .	29

8.6.1. Normas FIPS . . . . .	29
<b>9. Lenguaje de programación y bibliotecas utilizadas</b>	<b>30</b>
<b>10. Metodología de trabajo</b>	<b>30</b>
<b>11. Implementación de Referencia</b>	<b>31</b>
<b>12. Conclusiones</b>	<b>32</b>
<b>13. Disclaimer</b>	<b>32</b>
<b>14. Trabajo futuro</b>	<b>33</b>
<b>Bibliografía</b>	<b>37</b>

## 1. Abstract

La autenticación multifactor combina credenciales de distinto tipo para crear una defensa por capas y prevenir accesos no autorizados, esto aumenta la seguridad pero disminuye la experiencia del usuario. En este trabajo proponemos un sistema que provee autenticación multifactor utilizando “algo que tengo” y “algo que soy”, mediante un dispositivo de autenticación biométrica. Este sistema está orientado a proveer un servicio a empresas que ya utilicen la autenticación mediante un secreto compartido como una contraseña y busquen aumentar su seguridad.

Para realizar esto llevamos a cabo una comparación de distintos dispositivos biométricos de bajo costo para el desarrollo de aplicaciones de IoT. Si bien estos dispositivos no poseen la resolución adecuada para extraer características que permitan distinguir correctamente a los usuarios, se pudo implementar un esquema de autenticación mediante señales de electromiografía generadas con participación activa de los mismos.

## 2. Introducción

En la actualidad, internet se ha vuelto una herramienta fundamental en casi todos los aspectos del ser humano. Hace ya mucho tiempo que dejó de ser tan solo un método de comunicación conveniente. Por ejemplo, las personas ahorran mucho tiempo y dinero comprando en sitios e-commerce o haciendo trámites online. También, se entretienen con juegos, plataformas de música y video y con las redes sociales. Sin embargo, la mayoría de sitios web siguen requiriendo a los usuarios que se registren utilizando un nombre de usuario y una contraseña, un método que sigue sobreviviendo desde el comienzo de internet.

No todos los sitios de internet toman los recaudos necesarios en materia de seguridad para almacenar las contraseñas o para no permitir que atacantes intenten adivinar las mismas. Esto puede suceder por desconocimiento o porque los sistemas quedan desactualizados y sin mantenimiento o por vulnerabilidades de seguridad, entre otros.

Otro de los problemas los usuarios por lo general suelen olvidar las contraseñas muy largas y complejas (lo que hace que las contraseñas sean más seguras). Para contrarrestar este último punto, muchos sitios obligan a las personas a generar contraseñas que contengan letras mayúsculas y/o símbolos, esto sirve para aumentar la cantidad total de posibles contraseñas, y así disminuir la posibilidad de éxito frente a un ataque. Si bien esto mejora la seguridad frente a una búsqueda aleatoria o por fuerza bruta, los atacantes realizan búsquedas más inteligentes utilizando contraseñas o palabras conocidas, transformaciones simples de las mismas y también información de los usuarios (si la tienen).

Para empeorar aún más la situación, los usuarios suelen repetir contraseñas entre los diferentes sitios para no olvidarlas, lo que hace que exista una fuga de información indirecta cuando un sitio web es hackeado. Actualmente se sabe que más de cinco mil millones de contraseñas fueron adquiridas por hackeos públicamente conocidos [Hun].

Mejorar la seguridad en internet y proteger la información de los usuarios es sumamente importante y para eso es necesario atacar los dos problemas mencionados: los sistemas y la experiencia de usuario.

El cuerpo humano posee una actividad eléctrica natural que puede utilizarse para transmitir información a un dispositivo electrónico. Con este objetivo, son de particular importancia la electromiografía y la electroencefalografía. En el pasado estas señales eran utilizadas únicamente para propósitos médicos, pero en los últimos años con el surgimiento de las Brain Computer Interfaces (BCI) aumentó la disponibilidad de dispositivos de uso comercial. A la vez con la surgimiento de dispositivos *wearables* aparecieron nuevas posibilidades para la aplicación de sensores biométricos. Las aplicaciones de ésta tecnología son muy variadas, van desde control de prótesis robóticas [Vee] hasta otras más tradicionales como el registro de actividad cerebral durante el sueño o meditación [Cha15].

Uno de los aspectos más interesantes de este tipo de señales es que, si se logran los avances tecnológicos necesarios, podría suceder que los usuarios no necesiten realizar una acción extra para usar estas señales en la autenticación de un sistema, ya que podrían obtenerse a través de dispositivos que usa habitualmente de forma automática, logrando una mejor experiencia de usuario.

### 3. Justificación del trabajo

En este trabajo se evaluará la factibilidad de la utilización de señales EMG o EEG para utilizar como método de autenticación o segundo factor. Es por eso que se investigará si existe información intrínseca de los individuos en dichas señales y de qué manera se pueden implementar para que utilizarlo en sistemas existentes sea lo más simple posible.

Este tipo de señales tienen las ventajas que se obtienen mediante métodos no invasivos y con una participación tanto pasiva como activa de los individuos, lo que ofrece flexibilidad para aumentaría la aceptación psicológica de los mismos.

En primer lugar evaluamos distintos dispositivos de EMG y EEG disponibles para su uso en un sistema de autenticación. Esto implicó implementar una interfaz para comunicarse con los mismos y poder determinar las ventajas y desventajas de cada uno. A partir de estos datos tomamos la decisión de quedarnos con uno para diseñar una implementación testigo en la que se pueda observar la factibilidad del sistema, finalmente presentamos una API para dar un uso comercial al mismo

Nuestro trabajo de divide en tres partes: primero se presentan los distintos tipos de señales biométricas disponibles y el estado del arte en su uso como factor de autenticación. Luego se comparan distintos dispositivos biométricos con el fin de decidir cual utilizar para una implementación y habiendo elegido uno se detalla el procesamiento de la señal obtenida. Para finalizar se presenta una implementación de una API genérica para realizar autenticación mediante una señal biométrica con un fin comercial.

## 4. Estado del arte

### 4.1. Electromiografía (EMG)

EMG es una técnica de electro diagnóstico médico para grabar la actividad eléctrica de los músculos esqueléticos (unidos al esqueleto) provenientes del sistema nervioso. Existen dos tipos de EMG superficial e intramuscular. EMG superficial consiste en colocar electrodos, pequeños discos metálicos de algún material altamente conductivo, sobre la piel. Generalmente se aplica alguna sustancia entre el electrodo y la piel para reducir la impedancia entre ellos, como por ejemplo alcohol. Este tipo de EMG permite una visión general del funcionamiento del músculo y tienen la ventaja de ser procedimientos no invasivos. En EMG intramuscular los electrodos son agujas que se insertan en la piel hasta el tejido muscular, esto permite un análisis más detallado (debido, principalmente, a la reducción de ruido en las señales) de la actividad en el músculo pero tiene la desventaja de ser invasivo.

Cuando se observan biopotenciales colocando electrodos sobre la piel, no se registran potenciales intracelulares sino potenciales extracelulares, que son la manifestación de los primeros en el exterior al imprimir corrientes sobre los tejidos circundantes, que ofician de conductor de volumen. Además, generalmente no se observan potenciales de acción individuales, sino la superposición de un gran número de éstos originados en distintas células, que se combinan en el mencionado conductor de volumen. La amplitud y la morfología de las señales observadas dependen de múltiples factores, como el tipo de células involucradas, la cantidad de ellas y la distancia a la superficie, entre otras. [Hab16]

Para este proyecto se consideraron únicamente dispositivos de EMG superficial.

### 4.2. Electroencefalografía (EEG)

EEG es otra técnica de electro diagnóstico médico que sirve para grabar la actividad eléctrica del cerebro. Así como con EMG puede o no ser invasiva dependiendo del nivel de detalle requerido. Las técnicas no invasivas son más frecuentes, ya que las invasivas requieren perforaciones en el cráneo por lo que son utilizadas en pacientes con condiciones severas. Existen aplicaciones comerciales de EEG superficial para el monitoreo del sueño y meditación como el Muse Headband [Figura 8].

En general, debido a que las mediciones se realizan con electrodos en el cuero cabelludo, difícilmente se puedan obtener buenas mediciones de la actividad directa del Sistema Nervioso Central y son fuertemente dependientes de la ubicación de los electrodos. Además, si bien es posible detectar de manera general las áreas del cerebro que se encuentran en actividad, la localización espacial es bastante pobre y es muy difícil asociar qué acciones específicas producen las señales.

Otro aspecto importante es que los dispositivos con mejor resolución, además de ser muy costosos, atentan contra la usabilidad en una posible implementación comercial, debido a que lleva mucho tiempo colocarse los electrodos en la cabeza de forma correcta.

Es por esto que en dispositivos como el Muse, que es muy simple y fácil de colocar, se pierde resolución y calidad de la señal.

#### 4.3. Sistema de Autenticación por Biometría

La autenticación es un proceso por el cual se asocia una entidad externa a una representación interna de dicha entidad en el sistema. Los pasos que se requieren para realizar una autenticación son: obtener información de la identidad, analizar dicha información y verificar si corresponde con alguna representación. La información requerida suele tener diferentes grados de confianza y proveer de diferentes fuentes, y las mismas se suelen categorizar en:

- Algo que conozco: esta es la fuente más utilizada y se basan en secretos compartidos, como puede ser una clave que solo el usuario conoce.
- Algo que tengo: basan la identificación en algún elemento físico que genera claves pseudo aleatorias.
- Algo que soy: basan la identificación en alguna información intrínseca, única e inalterable del individuo, entre las más conocidas se encuentran las huellas digitales o el iris. Los métodos no son 100 % eficaces.
- Contexto: esta fuente se refiere a información circunstancial del usuario como puede ser su IP, la computadora que está utilizando, etc.

Utilizar más de una vez una misma fuente, por ejemplo requerir dos contraseñas, agrega más seguridad al sistema, sin embargo, es muy probable que un atacante que logre obtener una pieza de información secreta de un usuario de un tipo de fuente, también pueda lograr obtener el resto de la misma forma. Es por eso que los sistemas más seguros se obtienen al combinar diferentes fuentes, por ejemplo, pidiendo un secreto compartido, un numero aleatorio generado por un dispositivo criptográfico físico y además comparando la información de contexto con los patrones de uso previos registrados.

La fuente más utilizada y simple de implementar es aquella que utiliza un secreto compartido (usuario y contraseña). Históricamente agregar una fuente extra resultaba muy costoso tanto en lo económico como en la experiencia del usuario. En los últimos años se desarrollaron varias técnicas y tecnologías que permitieron disminuir estos costos y sumado a que existe una mayor concientización por parte de los usuarios, se comenzaron a utilizar las otras fuentes de identificación como un segundo factor. Entre los ejemplos más comunes se encuentran: lector de huella en los celulares, códigos enviados por SMS o email y aplicaciones que generan números pseudo aleatorios en los celulares. Agregar fuentes siempre atenta contra la experiencia de usuario, ya que requieren mayor participación del mismo y hacen que acceder al sistema tarde más y requiera más pasos.

Un descubrimiento reciente demostró que la señal proveniente de los latidos del corazón es única en los individuos, lo que resulta de mucho valor en la industria y por ejemplo Samsung ya aplicó una patente en Estados Unidos para el uso de la señal del corazón (ECG) para el desbloqueo de sus relojes inteligentes [Yan16].

La autenticación biométrica se basa en la autenticación de los usuarios en función de sus características físicas y consiste en la aplicación de técnicas matemáticas y estadísticas sobre los rasgos físicos de la persona con el objetivo de verificar su identidad. Básicamente, en un primer registro de alta, se “escanea” la característica biométrica que es analógica, mediante un algoritmo se digitaliza, se asocia a la identidad personal y se guarda de forma cifrada. Cuando se autentica el individuo, se compara la información recogida con el repositorio de características biométricas digitalizadas y se determina la identidad. [Esp17]

En los últimos años se publicaron varios papers de autenticación por biometría como *A Study on EMG-based Biometrics* [KP17] que propone un esquema de autenticación que utiliza señales de EMG obtenidas realizando tres movimientos: cerrar el puño, rotar la muñeca y levantar la mano. También existen esquemas que utilizan otro tipo de señales como *A novel biometric authentication approach using ECG and EMG signals* [Bel+15] que esta basado en señales ECG pero también agrega una componente de EMG. En estos papers se utilizan dispositivos de aplicación médica, los cuales proveen una mejor calidad de señal, o trabajan sobre bases de datos médicas cuyos datos fueron obtenidos con dispositivos de este tipo. El esquema propuesto en este trabajo utiliza dispositivos no pensados para uso médico lo que implica un menor costo pero a la vez una peor calidad de señal.

## 5. Dispositivos para Captura de Señales Biométricas

En esta sección evaluamos los dispositivos: TrueSense, Olimex EMG/EEG Shield y Muse Headband, tomando en consideración las señales que permiten obtener y su facilidad de uso y desarrollo.

### 5.1. TrueSense

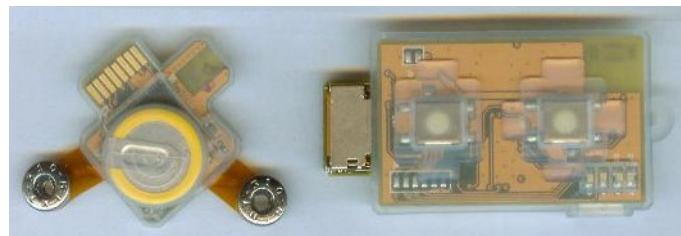


Figura 1: Sensor y controlador TrueSense

El TrueSense es un dispositivo para leer señales EMG producido por Open Path Innovations, el cual se consta de un sensor y un controlador. El sensor posee una batería y le envía la señal al controlador de forma inalámbrica, lo que lo hace más cómodo y práctico de utilizar que otras alternativas, es por eso es que fue nuestra primera opción para comenzar el armado del prototipo.

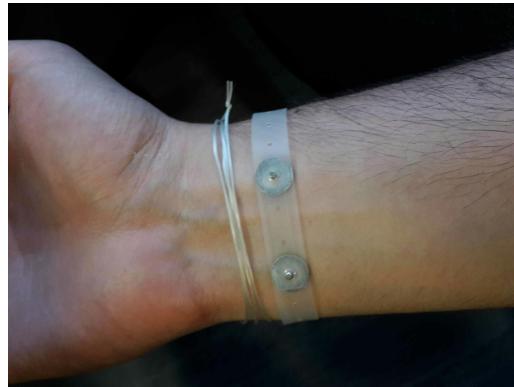


Figura 2: Ubicación de los electrodos en la muñeca



Figura 3: Ubicación de los electrodos en el antebrazo

**El sensor se conecta a electrodos que se ajustan con una banda elástica en cualquier zona del cuerpo.**

#### 5.1.1. Problemas encontrados

La empresa fabricante del dispositivo decidió discontinuar el producto, por lo tanto todos los drivers y documentación quedaron sin mantenimiento alguno. El código que se proveía en el SDK del dispositivo estaba realizado en C++ y debido a que no conocemos en profundidad ese lenguaje, nos iba a retrasar en el armado del prototipo. Es por esto que se decidió construir un driver nuevo en Python usando la biblioteca pyserial y la documentación provista.

La documentación provista por la empresa en su sitio web oficial, pertenece a una versión del dispositivo anterior a la que tuvimos acceso, por lo tanto se utilizó el código C++ como referencia de las partes no documentadas o desactualizadas.

Durante las pruebas del dispositivo nos encontramos con que la batería del sensor tiene una duración de aproximadamente 20 minutos, además de tener que reiniciar la captura de señales por errores de calibración del dispositivo. Estos motivos hicieron que no se pudieran tomar muestras de forma consistente.

Se tomaron muestras de distintos movimientos, entre ellos: cerrar y abrir la mano, flexionar muñecas y mover individualmente los dedos. La idea que se tenía era poder diferenciar dichos movimientos para que los usuarios pudieran crear contraseñas más complejas.

Para probar la efectividad de la clasificación, en principio, se tomaron muestras que sólo contenían un único movimiento, para luego, en caso de una clasificación satisfactoria, probar si era posible detectar los movimientos en una señal compuesta por varios movimientos.

Para clasificar los movimientos, se extrajeron las features para series de tiempo recomendadas en el paper [Ang09]:

Integrated EMG	$IEMG = \sum_{n=1}^N  X_n $
Mean absolute value	$MAV = \frac{1}{N} \sum_{n=1}^N  X_n $
Simple Square Integral	$SSI = \sum_{n=1}^N  X_n ^2$
Variance of EMG	$VAR = \frac{1}{N-1} \sum_{n=1}^N X_n^2$
Root mean square	$RMS = \sqrt{\frac{1}{N} \sum_{n=1}^N X_n^2}$
Waveform length	$WL = \sum_{n=1}^{N-1}  X_{n+1} - X_n $

En las [Figura 4], [Figura 5] y [Figura 6] se pueden observar los resultados obtenidos de 20 muestras de los movimientos de cerrar dedos individualmente para distintas features.

Referencias de nombres de los dedos en las figuras	
Finger 0	Dedo pulgar
Finger 1	Dedo índice
Finger 2	Dedo medio
Finger 3	Dedo anular
Finger 4	Dedo meñique

En los gráficos de las features extraídas, a simple vista, surge como hipótesis que no se pueden clasificar correctamente los movimientos.

Se pensó en utilizar una red neuronal para clasificar los movimientos, pero debido a no poder contar con un set grande de muestras (ni poder tomarlo nosotros mismos), la clasificación no hubiese resultado satisfactoria y por eso se descartó ese método.

Para poder probar la hipótesis, se probó utilizar el clasificador naive de Bayes cuya fórmula es:

$$WL = \sum_{n=1}^{N-1} |X_{n+1} - X_n|$$

## Feature: mean\_absolute\_value Movimiento: cerrar dedo

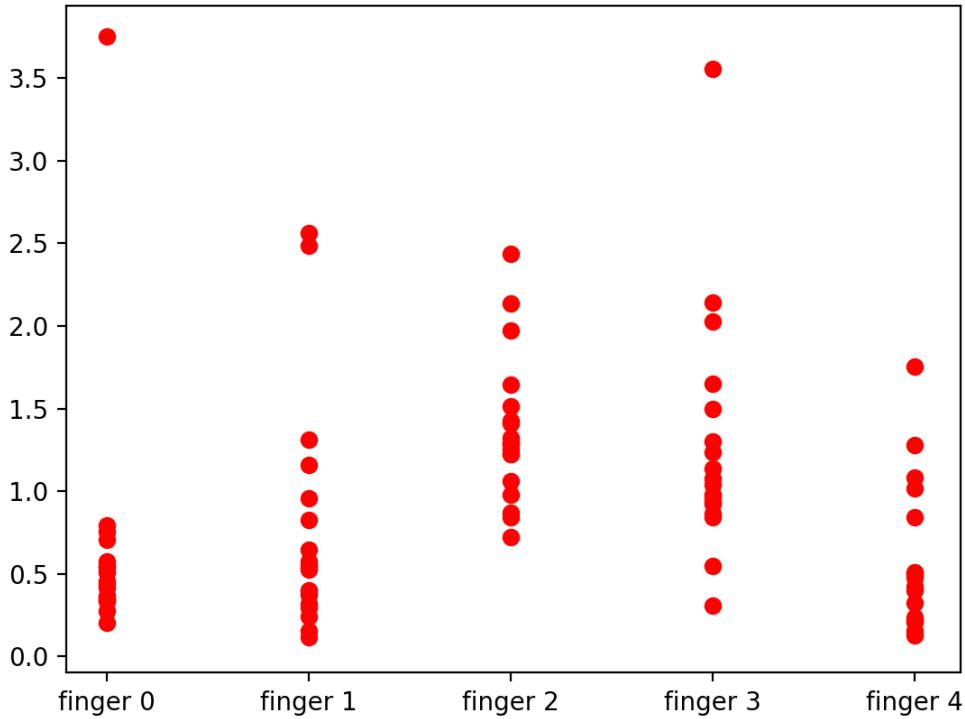


Figura 4: Feature “mean absolute value” para los movimientos de cada dedo

Para probar, se utilizó el set completo de cada dedo y se probó contra los distintos sets, lo que arrojó los siguientes resultados (sólo se muestra la feature “mean absolute value” porque fue la que arrojó los mayores porcentajes de clasificación):

Entrenamiento / Prueba	Finger 0	Finger 1	Finger 2	Finger 3	Finger 4
Finger 0	<b>34.22 %</b>	30.17 %	11.51 %	15.86 %	34.07 %
Finger 1	40.12 %	<b>35.18 %</b>	12.81 %	18.08 %	39.64 %
Finger 2	81.25 %	70.50 %	<b>34.99 %</b>	47.88 %	71.07 %
Finger 3	52.73 %	47.78 %	31.15 %	<b>36.22 %</b>	50.67 %
Finger 4	22.22 %	19.91 %	1.65 %	4.39 %	<b>26.08 %</b>

Idealmente, para poder clasificar los movimientos, lo que se debería ver en la tabla es que los valores de la diagonal estén cerca del 100 % y el resto de los valores estén lejos de un 100 %, pero como podemos ver los resultados no fueron así y la hipótesis de que no se puede clasificar correctamente se confirma.

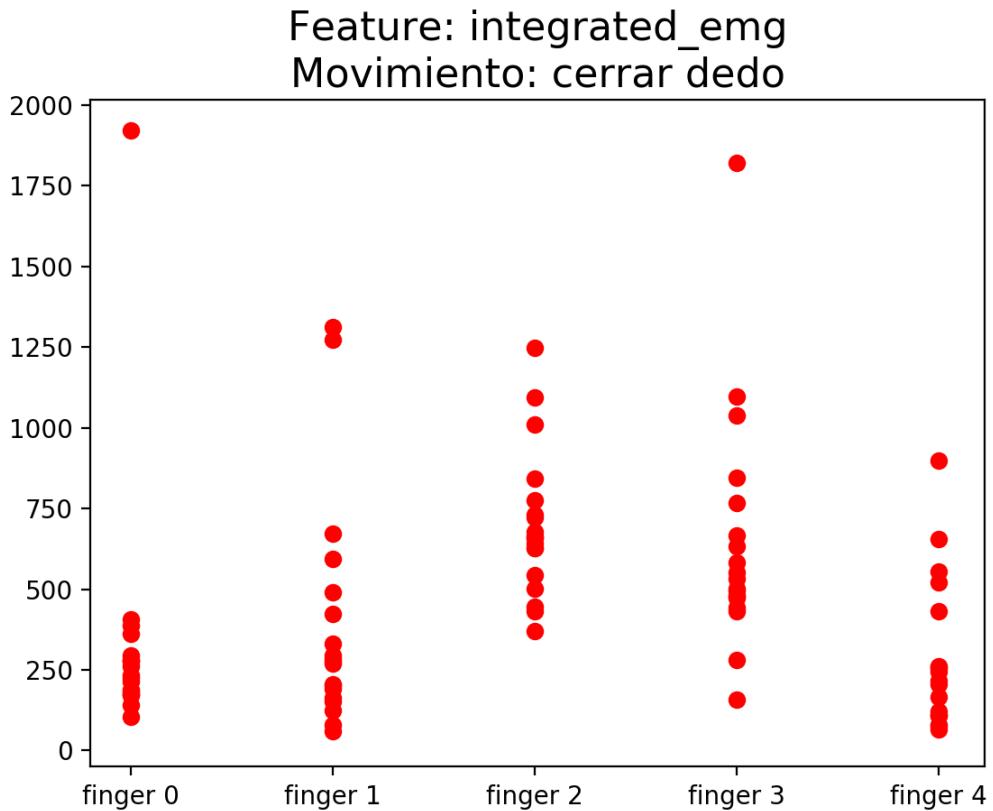


Figura 5: Feature “integrated emg” para los movimientos de cada dedo

Uno de los objetivos que debe tener un esquema de autenticación para conseguir la aceptación psicológica de los usuarios es que debe funcionar con poco entrenamiento por parte del mismo, y dado que no se pudo conseguir, decidimos utilizar las señales para la creación de contraseñas por parte de los usuarios y utilizar un dispositivo que nos permita tomar las muestras de forma más precisa y consistente.

## 5.2. Olimex EMG/EEG Shield

El Olimex EMG/EEG Shield [Figura 7] es una placa diseñada por la empresa Olimex que permite leer señales EMG o EEG desde un Arduino. Para hacer esto se deben conectar los electrodos deseados a la placa y la placa al Arduino. La placa se encarga de convertir la señal diferencial analógica (bioseñal) de los canales de entrada positivo y negativo, en un stream de datos como salida. Este stream puede ser leído por el Arduino mediante los pines analógicos. El Arduino se encarga de discretizar la señal y escribirla en el puerto serie.

Las ventajas de este dispositivo frente al TrueSense son:

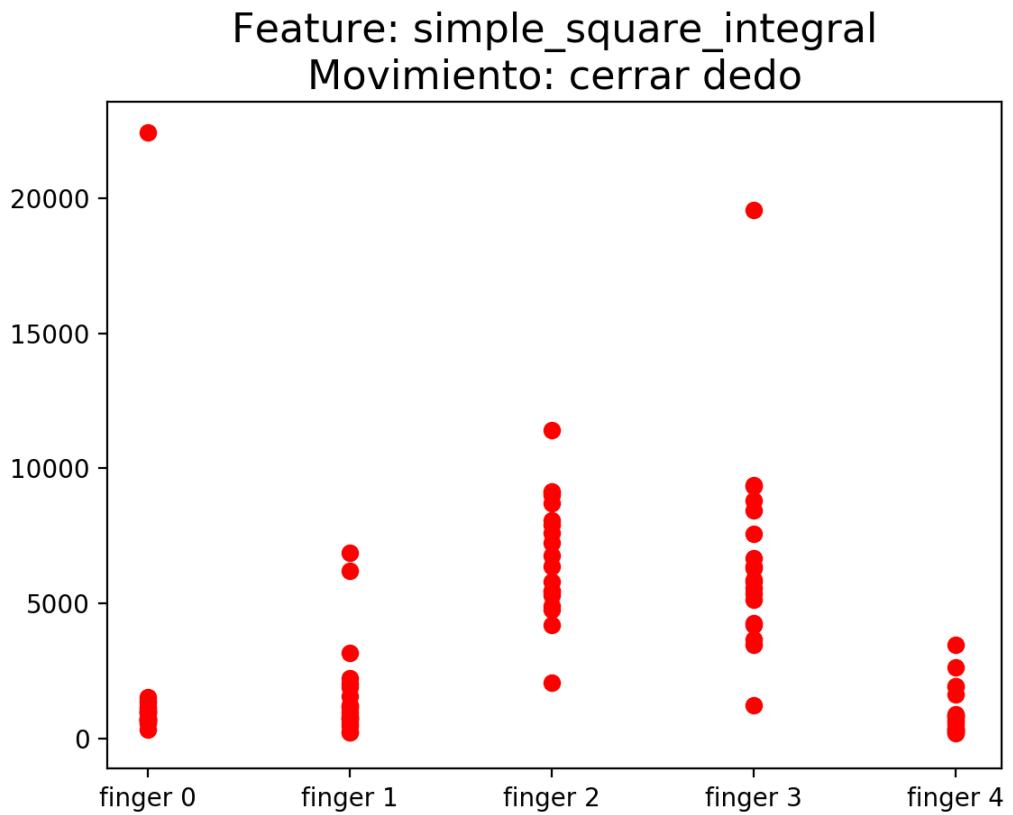


Figura 6: Feature “simple square integral” para los movimientos de cada dedo

1. Al no tener baterías, es mucho menos tedioso realizar pruebas que se prolonguen en el tiempo.
2. Debido a que se puede conectar cualquier electrodo, utilizamos unos que se adhieren mejor a la piel y tenían menos ruido.
3. La lectura por el puerto serie es mucho más simple.
4. Se puede definir la frecuencia de muestreo.

### 5.3. Muse headband

Este es un dispositivo pensado para el monitoreo señales de EEG durante la meditación [Figura 8]. El dispositivo se conecta mediante Bluetooth LE (4.0) con un teléfono que tenga la aplicación de Muse, desde esta se pueden ver distintas métricas de las señales.

También se encuentra disponible para descargar un SDK para iOS, Android y Windows 10 y del sitio de Muse también se puede descargar un proyecto de Visual Studio 2015 con una aplicación para visualizar la señal.



Figura 7: Fotografía de un Olimex EMG Shield



Figura 8: Imagen de un Muse headband

El Muse presentaba una serie de ventajas como ser fácil de colocar y una buen duración de la batería. Pero como desventajas tiene que, debido a que fue diseñado para ser utilizado desde un dispositivo móvil, la comunicación desde una computadora es dificultosa. Las aplicaciones para comunicarse con MacOS y Linux quedaron obsoletas con el Muse 2016 que fue el provisto, únicamente soportando Windows 10. A su vez también es necesario que la computadora cuente con un conector Bluetooth LE.

#### 5.4. Conclusión

De los dispositivos disponibles decidimos quedarnos con el Olimex Shield ya que el mismo supera ampliamente al TrueSense tanto en la calidad de la señal como en la facilidad de aplicación de los electrodos. El Muse presentaba una alternativa interesante por ser ya un producto de uso comercial, lo que se puede observar en su apariencia y facilidad de uso, pero fue descartado debido a la dificultad para conectarse desde otro dispositivo más allá de un smartphone.

## 6. Análisis de la señal EMG

Como se mencionó anteriormente, debido a la dificultad de obtener las señales EMG del dispositivo TrueSense y las señales EEG del Muse, en esta sección se describirá como fue el análisis de las señales obtenidas por el Olimex, que son las que fueron utilizadas.

El análisis de la señal consiste en dos partes, el preprocesamiento y la codificación. La señal cruda que proviene del dispositivo consiste en la diferencia de potencial entre los electrodos derecho e izquierdo. Con los electrodos colocados sobre un músculo en particular, los picos en la señal reflejan la activación del mismo.

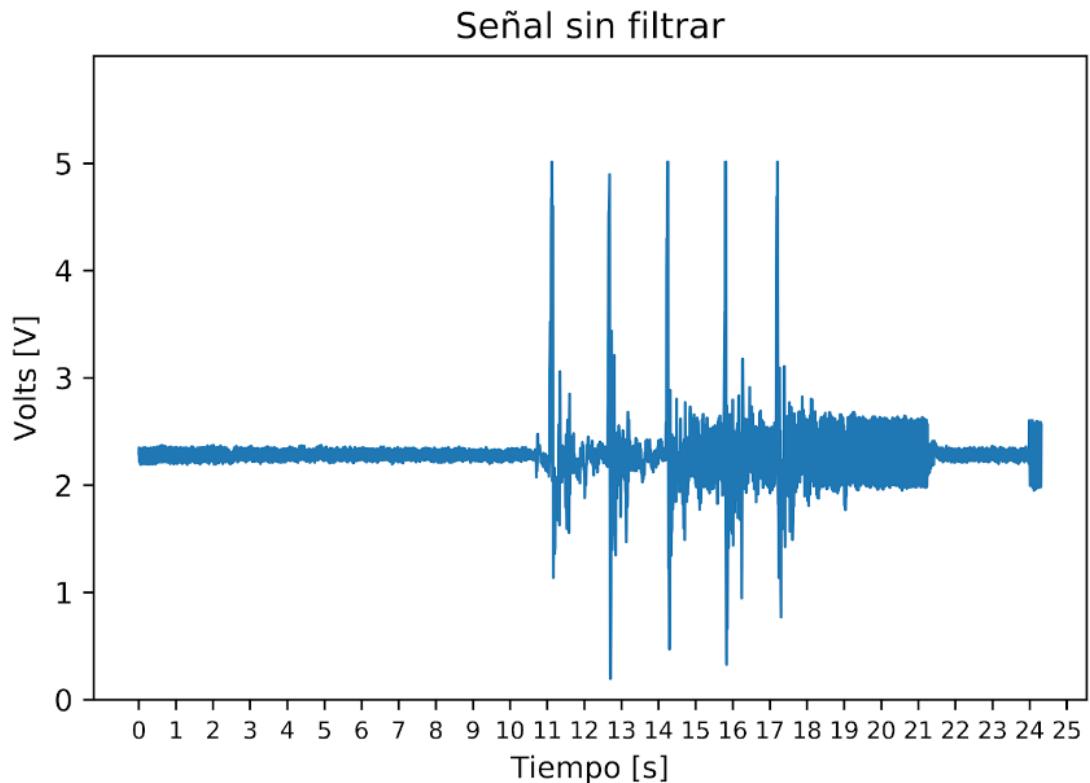


Figura 9: Ejemplo de una señal no filtrada obtenida

El en Olimex Shield, el rango de valores de la señal es de 0 a 5 volts mapeado a valores enteros de 0 a 1023 y la frecuencia de muestreo es de 256 Hz.

El objetivo es intentar encontrar picos en la señal, es decir, momentos en los que la persona haya realizado algún movimiento que produjo un cambio de potencial.

La señal sera dividida en partes o *ventanas* de longitud N configurables. Como el dispositivo tiene una frecuencia de muestreo de 256 Hz, una ventana de 256 valores representa la señal en un período de 1 segundo. El valor que se le da a N es importante ya que afecta directamente a la experiencia de usuario: un valor grande hará que la persona tenga un control más preciso pero que la duración del proceso sea más lento, mientras que un valor muy pequeño hará que el usuario pierda precisión y aumente la velocidad del proceso. Luego se aplicará el siguiente filtro a cada ventana:

$$f(s, i, n) = \max(\{s(x) : x = i - n, \dots, i\})$$

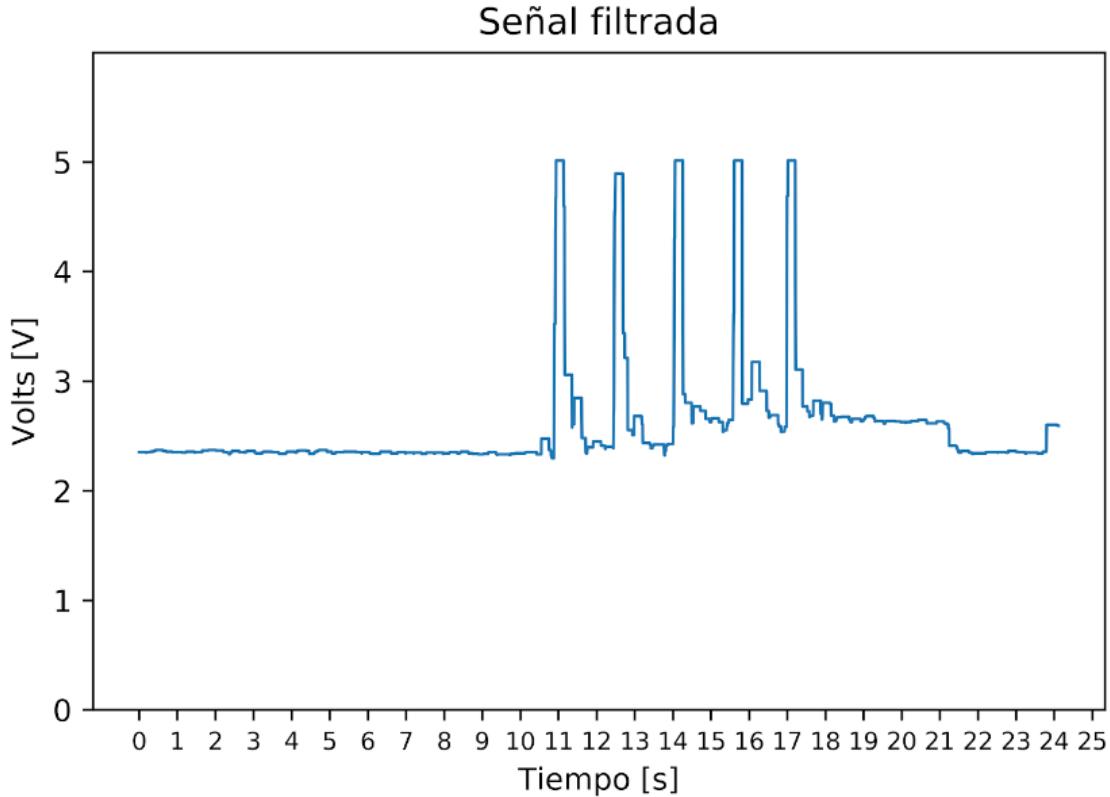


Figura 10: Resultado de filtrar la señal de la figura 9. Se puede observar menos ruido y se aprecian mejor los picos de activación. La ventana utilizada fue de 50 muestras, que con una frecuencia de muestreo de 256 Hz da una duración de aproximadamente 0.2 segundos.

**El resultado visible de este filtro es que se logra una especie de señal cuadrada”, ya que nos quedamos con el máximo valor de cada una de las ventanas [Figura 10].**

La segunda etapa de análisis consiste en transformar la señal a un código binario, siendo 1 un pico de activación y 0 el reposo. Para esto se debe elegir un valor de umbral, el cual hace que todos los valores por debajo del mismo se transformen en 0 y todos los valores superiores se transformen en 1, de esta manera detectar un cambio se vuelve muy simple y podemos generar el código binario [Figura 11].

### 6.1. Comparación de las señales

Una vez transformadas las señales en cadenas de bits, lo que nos queda es conseguir una manera de diferenciar dos señales.

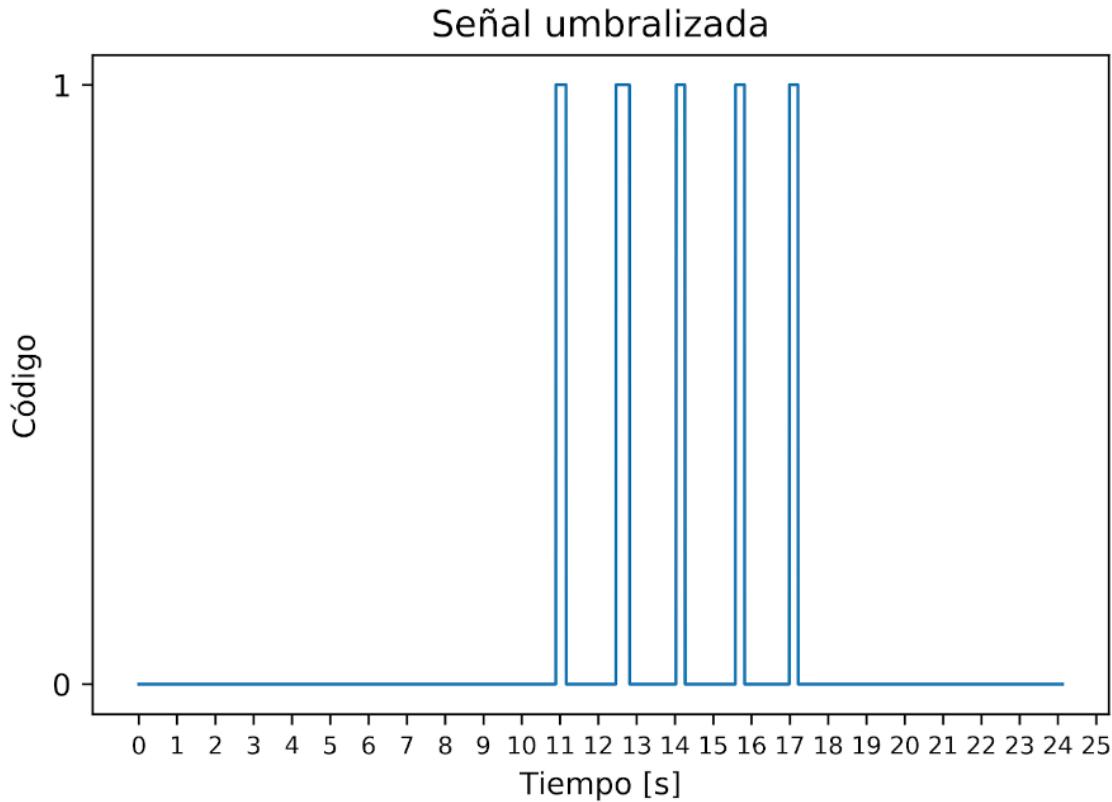


Figura 11: Resultado de umbralizar la señal de la figura. El umbral elegido fue de 650 lo que serian unos 3.185 volts. 10

Dado que la señal ahora es binaria, se puede utilizar cualquier método existente en las telecomunicaciones y se eligió utilizar la distancia de Hamming. Esta se define como el número de bits que tienen que cambiarse para transformar una palabra de código válido en otra palabra de código válida.

Para calcular dicha distancia hay que sumar la cantidad de bits que difieren de una palabra respecto a la otra. Luego definimos la diferencia entre dos señales, siendo  $B$  la cantidad de bits diferentes y  $L$  las longitudes de las señales, como sigue:

$$D(L_1, L_2, B) \begin{cases} 0 & , si L_1 \neq L_2 \\ 1 - \frac{B}{L_1} & si, L_1 = L_2 \end{cases}$$

## 7. BAD API

Agregar un multifactor que utilice bioseñales a un sitio existente no es algo simple de hacer, por eso es una buena idea utilizar un servicio externo que permita hacerlo de la

forma más simple posible.

Luego de la investigación realizada, se decidió implementar un esquema testigo completo de autenticación por segundo factor basado en señales de EMG. La solución incluye un servicio SaaS que provea una forma de agregar una autenticación reutilizable desde cualquier tipo de aplicación externa. A este servicio lo llamamos BAD API (Biosignal Authentication Device) y la idea es que sea un servicio Cloud Platform As A Service que puedan utilizar clientes en sus aplicaciones. Los clientes de este servicio no son los usuarios finales de un sistema, sino que son, en principio, los proveedores de servicios web (por ejemplo el home banking de un banco) que deseen agregar este tipo de segundo factor para sus usuarios.

El servicio BAD API proveerá a los clientes de los dispositivos que leen las señales EMG y cuenta con un servidor en la nube que los mismos pueden utilizar para conectarse. Para esta conexión es necesario un access key que se proveerá a los clientes junto con los dispositivos.

En los siguientes apartados se explicarán todos los detalles y fundamentos técnicos del servicio.

### 7.1. Arquitectura

El servicio cloud utilizado para desplegar la BAD API y para comunicar de forma segura los dispositivos fue Google Cloud Platform, mas precisamente sus servicios IoT Core [Gooa], App engine [Goob] y Cloud SQL [Gooc].

IoT Core es un servicio que permite administrar dispositivos IoT y que los mismos se comuniquen de forma segura. Cada dispositivo debe tener un par de claves pública y privada y debe estar registrado en la consola web que provee Google con su clave pública. Los dispositivos que no utilizan una clave registrada tienen el acceso denegado.

### 7.2. Comunicación

Los dispositivos envían las señales y el estado de los mismos utilizando el protocolo MQTT y enviando la información de forma encriptada utilizando TLS. Este protocolo permite crear canales de comunicación en los que los agentes pueden publicar y otros pueden suscribirse. En este caso los dispositivos publican y la BAD API se suscribe para recibir la información y guardarla en la base de datos.

El servicio IoT Core de Google permite a los suscriptores de los canales poder obtener información sobre los autores de los mensajes de manera automática, por lo tanto, no es necesario enviar esa información en los mensajes.

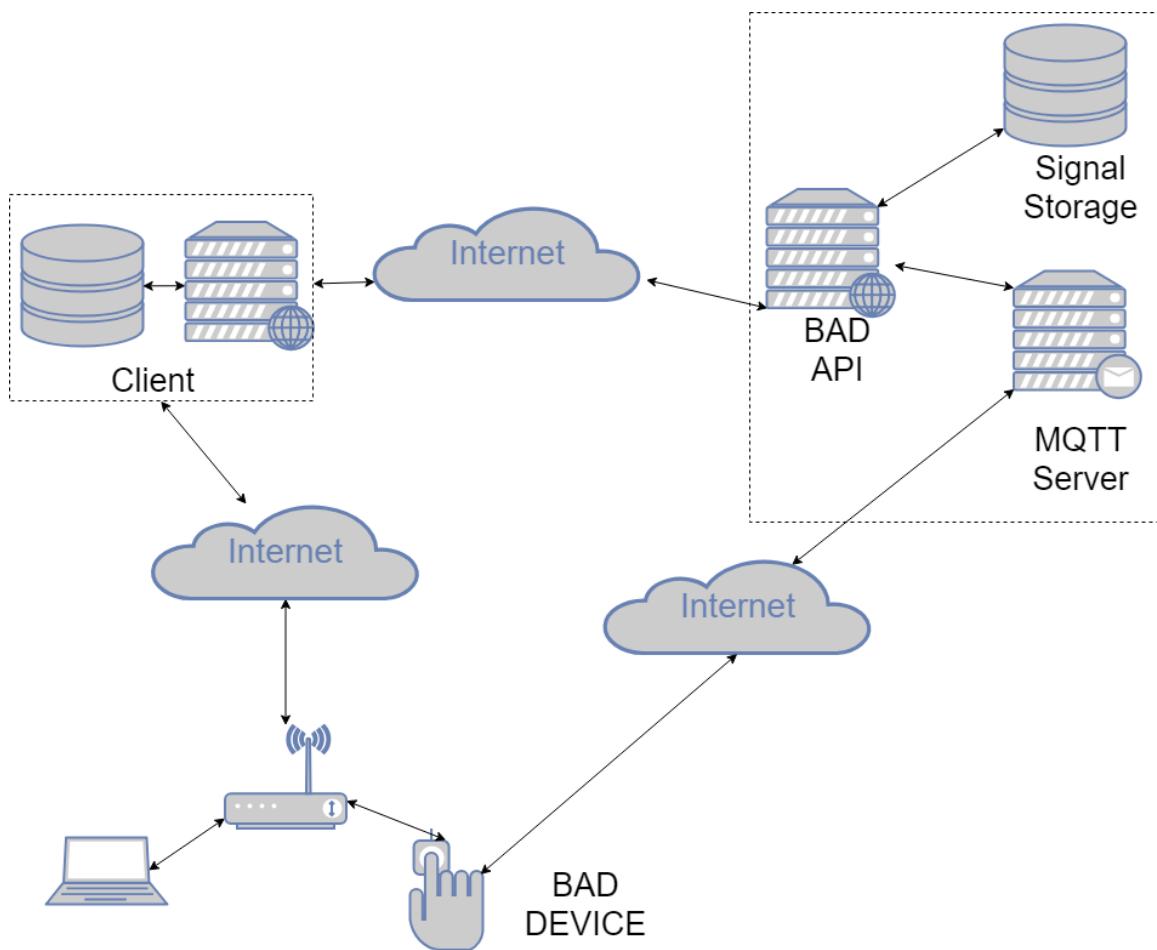


Figura 12: Diagrama general

### 7.3. MQTT

El protocolo de comunicación utilizado para la comunicación entre los dispositivos y el servicio cloud es el MQTT (Message Queuing Telemetry Transport). Es un protocolo del tipo *publish-subscriber* que funciona sobre TCP/IP.

Para comunicarse es necesario que exista al menos una cola de mensajes o canal.<sup>a</sup> la cual un agente se suscribe y otro publica. El protocolo en si mismo es simple y agrega muy poco *overhead* a los mensajes, por lo tanto el tamaño del mensaje es casi en su totalidad la información que se desea enviar. Esto último es principalmente útil en nuestro caso dado que sera utilizado por dispositivos IoT que tienen poco poder de procesamiento y probablemente conexiones a internet con poco ancho de banda.

El servicio IoT Core de Google permite a los suscriptores de los canales poder obtener información sobre los autores de los mensajes de manera automática, por lo tanto, no

es necesario enviar esa información en los mensajes.

#### 7.3.1. Canal “Status”

Uno de los aspectos que le interesan a la BAD API es la IP que tienen los dispositivos, ya que los usuarios se comunicarán con los mismos mediante la interfaz web expuesta en dicha IP. Para esto, existe un canal del protocolo MQTT exclusivo llamado “status” y tiene como objetivo enviar información del estado del dispositivo. En esta primera versión únicamente se utiliza para enviar la dirección IP de los dispositivos. En implementaciones futuras se podría agregar más información útil para detección de errores. El mensaje actual es el siguiente:

```
{  
    "IP": "192.169.0.11"  
}
```

#### 7.3.2. Canal “Signal”

Este canal se utiliza para enviar la señal a la BAD API. El mensaje actual es el siguiente:

```
{  
    "uuid": "9583cd17-9f85-4eac-8099-17fb3ab203b7",  
    "Signal": [0, 1, 0, 1, 1, 0]  
}
```

### 7.4. Comunicación entre los usuarios, clientes y la BAD API

El protocolo de comunicación principal elegido fue HTTP y HTTPS, lo que facilita la integración por parte de los clientes. Actualmente la conexión entre el cliente y la BAD API se utiliza con HTTPS con el certificado de Google mientras que la comunicación entre el usuario y los dispositivos se realiza mediante HTTP debido a que no contamos con un certificado propio de la BAD API aún. Una vez obtenido un certificado válido se utilizará para ambas comunicaciones y se dispondrá de un dominio propio para acceder a la API.

En principio, la aplicación cliente le muestra al usuario, las opciones de dispositivos que tiene disponibles para él [Figura 13]. Dado que el cliente tiene toda la información de sus dispositivos, este puede mostrar a sus usuarios la información inteligentemente (podría mostrarle sólo los dispositivos cercanos).

El usuario elige uno de los dispositivos disponibles y puede pedirle que comience a leer la señal, obteniendo un UUID que la representará en la BAD API.

Cuando el dispositivo recibe la señal de que pare de leer la señal, éste pública la misma en el canal MQTT provisto por el servicio cloud, el cual será recibido por la BAD API.

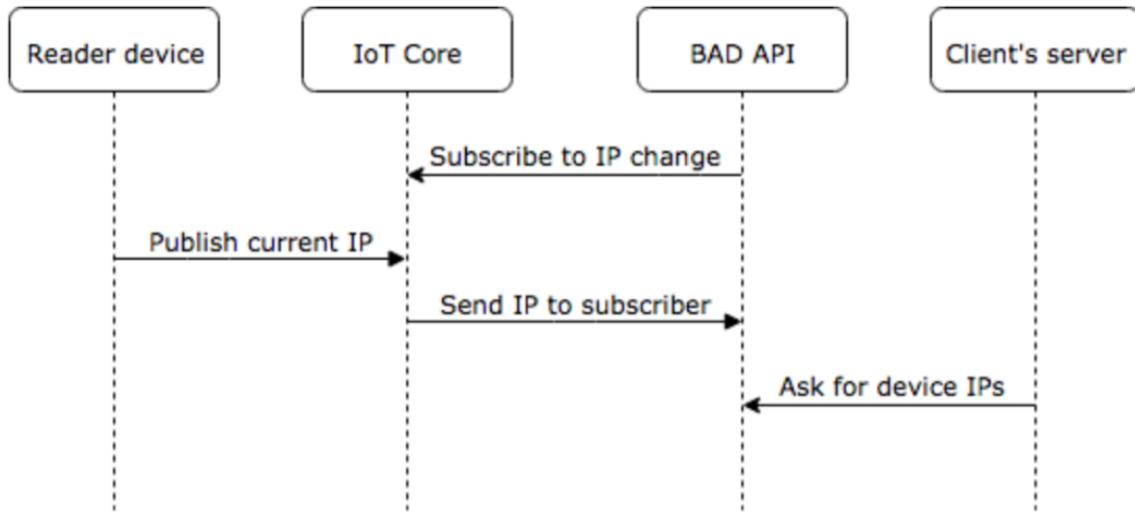


Figura 13: Flujo de comunicación de la IP del dispositivo

Una vez que la señal fue recibida por la BAD API, el usuario le envía al servicio cliente un request para registrarse o loguearse [Figura 14].

Es aqui el unico punto en donde difieren ambos flujos:

- Para el caso del registro, el servicio cliente solo verifica que la señal exista en la BAD API y no haya sido utilizada por algún otro usuario, en caso afirmativo el cliente guarda la información del usuario y el UUID de la señal con la que se registró.
- Para el caso del login, el cliente envía el UUID de la señal recién obtenida y el UUID de la señal de registro que tiene en su sistema a la BAD API para que las compare. Este último le envía un porcentaje de similitud entre ambas señales. El servicio cliente decide que hacer dado el porcentaje recibido (podrá denegar el login si es muy bajo o mostrar parte del contenido).

## 7.5. BAD-DEVICE-I: Prototipo del dispositivo

Para el primer prototipo se decidió utilizar dispositivos electrónicos de uso general (esto permite realizar modificaciones del prototipo de forma rápida y económica). Se utilizó una Raspberry PI a la cual se conectó un Arduino Chipkit Uno32 con un Shield EMG de Olimex, el cual se conecta a un sensor EMG de 3 electrodos: derecho (R) izquierdo (L) y diferencial (D). La señal obtenida consiste en la diferencia de potencial entre los electrodos derecho e izquierdo tomando como tierra el diferencial.

Para mejorar la calidad de la señal es recomendable aplicar alcohol sobre la piel en el lugar donde se sitúan los electrodos ya que esto reduce la impedancia de la piel.

La Raspberry tiene instalado un servidor web que expone una API JSON que permite a los clientes comunicarse fácilmente mientras que sus usuarios se encuentren en la

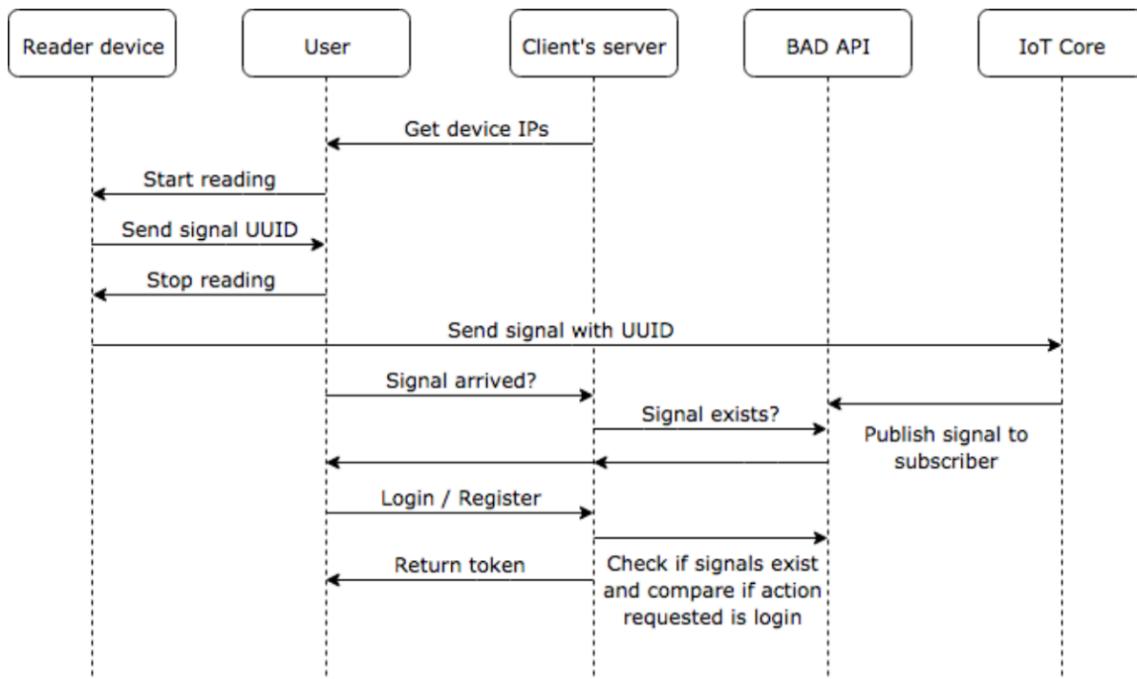


Figura 14: Esquema de autenticación utilizando el servicio de BAD API

misma red. En caso de exponer la Raspberry con una IP publica, el dispositivo podría ser accedido desde cualquier lugar.

Los parametros de la umbralización de la señal son configurables por el usuario. Esto permite que la lectura de la señal se adapte a cada usuario, lo cual es muy importante debido a las variaciones que se producen debido a la impedancia de la piel es diferente en cada individuo.

Además, la Raspberry posee el software y la clave privada necesarios para comunicarse con la BAD API a través de internet de forma segura. Dicha clave estará preconfigurada antes de entregar el dispositivo al cliente y la clave pública asociada estará registrada en el servicio web.

## 7.6. Integración por parte de los clientes y kickstart

Los clientes del servicio BAD deberán primero obtener un access key para comunicarse con la API. Estas keys se proveerán de forma directa luego de la contratación del servicio.

Una vez contratado el servicio, los clientes deben desarrollar una integración con las interfaces HTTP del dispositivo y de la API.

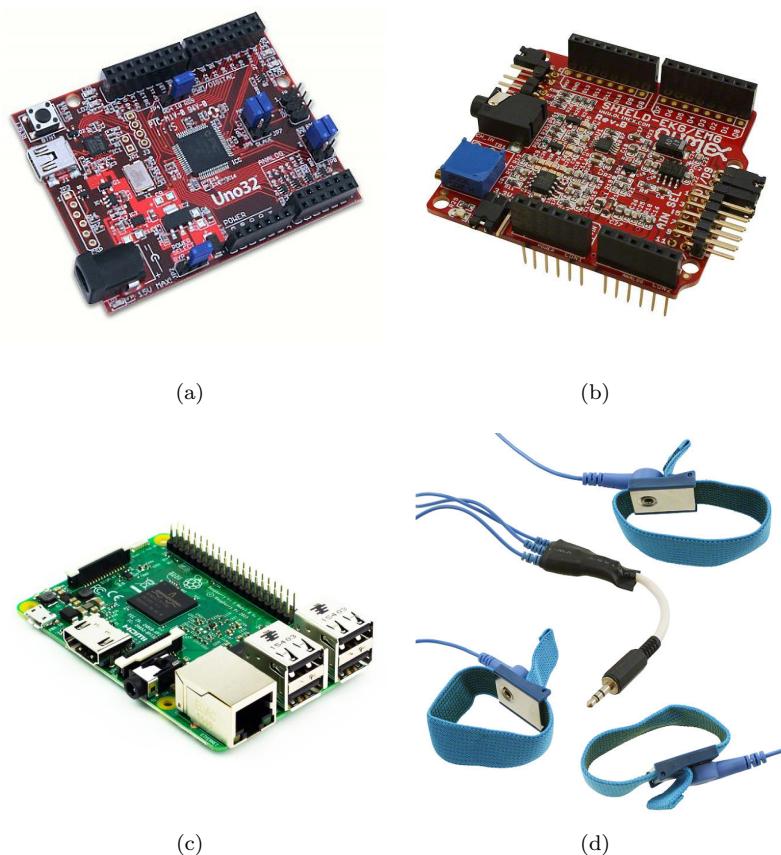


Figura 15: (a) Arduino Chipkit Uno32; (b) Olimex EMG/EEG Shield; (c) Raspberry Pi 3; y, (d) Electrodos EMG.

La documentación de ambas interfaces se detalla en el siguiente apartado. Tanto la documentación como una aplicación de prueba se mostrarán de forma pública para que los clientes puedan realizar la integración sin problemas.

## 7.7. Interfaz BAD API

Esta API expone una interfaz de comunicación mediante el protocolo HTTP y el formato de los mensajes intercambiados es JSON. El envío del access key se realiza mediante el header “Authorization” de HTTP.

### 7.7.1. GET /api/v1/signals/:uuid

Este endpoint sirve para verificar la existencia de una señal en la base de datos.

#### Parámetros de entrada

El único parámetro necesario es el uuid de la señal y se provee a través de la URL.

#### Salida

El cuerpo de la salida es vacío, la información está en el código HTTP de la respuesta.

#### Códigos de respuesta posibles

- 204: la señal se encuentra en la base de datos.
- 404: la señal no se encuentra en la base de datos o el dispositivo que la tomó no le pertenece al cliente que realiza el pedido.
- 401: No se envió el access token o no le pertenece a ningún cliente.

### 7.7.2. POST /api/v1/signals/compare

Este endpoint sirve para comparar dos señales pertenecientes a cualquier dispositivo de un cliente.

#### Parámetros de entrada

Este endpoint requiere los uuid de las señales en el body de la request HTTP:

```
{  
    "signal_1_uuid": "9583cd17-9f85-4eac-8099-17fb3ab203b7",  
    "signal_2_uuid": "7c4bd9df-e6f0-463d-aec5-ad3db7d815da"  
}
```

#### Salida

Se devuelve un valor entre 0 y 1 que representa la similitud de las señales.

```
{  
    "percentage": 0.834  
}
```

#### Códigos de respuesta posibles

- 200: el pedido fue satisfactorio.
- 404: alguna de las señales no se encuentra en la base de datos o el dispositivo que las tomó no le pertenece al cliente que realiza el pedido.
- 401: No se envió el access token o no le pertenece a ningún cliente.

#### 7.7.3. GET /api/v1/devices

Este endpoint le permite conocer a los clientes la información de sus dispositivos.

##### Salida

```
[  
    {  
        "id": "mac-2",  
        "ip_address": "192.168.0.179"  
    },  
    {  
        "id": "rpi-1",  
        "ip_address": "192.168.0.11"  
    }  
]
```

#### Códigos de respuesta posibles

- 200: el pedido fue satisfactorio.
- 401: No se envió el access token o no le pertenece a ningún cliente.

#### 7.8. Interfaz BAD-DEVICE-I

La interfaz expuesta en el prototipo también utiliza el protocolo HTTP y JSON como formato de los mensajes.

#### 7.8.1. POST /api/v1/start

Este endpoint le indica al dispositivo que debe comenzar a leer una señal y devuelve el UUID que identificará a la misma.

#### Salida

```
{  
    "signalUUID": "9583cd17-9f85-4eac-8099-17fb3ab203b7"  
}
```

#### Códigos de respuesta posibles

- 201: el pedido fue satisfactorio y se crea el UUID de la señal.

#### 7.8.2. POST /api/v1/stop

Este endpoint le indica al dispositivo que debe parar de leer la señal y enviará la señal a través del protocolo MQTT a la BAD API.

#### Salida

```
{  
    "signalUUID": "9583cd17-9f85-4eac-8099-17fb3ab203b7"  
}
```

#### Códigos de respuesta posibles

- 200: el pedido fue satisfactorio y se envió la señal a la BAD API.

#### 7.8.3. POST /api/v1/cancel

Este endpoint cancela la recolección de la señal actual sin enviar el resultado previo a la BAD API.

#### Códigos de respuesta posibles

- 200: el pedido fue satisfactorio y se cancela la recolección de la señal.

#### 7.8.4. GET /api/v1/read

Este endpoint permite a la aplicación cliente ver los valores que están siendo leídos por el dispositivo. No es necesario utilizar este endpoint para el correcto funcionamiento del sistema, pero sirve para dar un feedback a los usuarios.

#### Salida

Se devuelve tanto los valores de la señal cruda como los valores de la señal codificada. El formato de ambas señales es de un array que contiene las coordenadas X e Y de la señal.

```
{  
    "signal": [[0, 123], [1, 657], [2, 456]],  
    "Interpreted_signal": [[0, 0]]  
}
```

#### Códigos de respuesta posibles

- 200: el pedido fue satisfactorio.

## 8. Threat model

En esta sección se describirán las posibles amenazas y los aspectos de seguridad relevantes en el sistema, así también como fueron mitigados algunas de las amenazas y cómo se pueden mitigar las demás.

Hay cuatro canales de comunicación que manejan y/o almacenan información sensible [Figura 12].

### 8.1. Comunicación entre el dispositivo y la computadora del usuario

Como hemos descripto anteriormente, el usuario se comunica con el dispositivo para avisarle que debe comenzar, parar o cancelar la lectura de la señal y este le responde con el ID de la señal. Además se transmite la señal para que el usuario pueda recibir feedback del proceso.

Para que esta comunicación sea segura, se debe utilizar el protocolo HTTPS, por ende el dispositivo debe contar con un certificado X.509 firmado por una autoridad certificante que la computadora del usuario confíe. Como la comunicación hacia la BAD API también debería utilizar TLS, se pueden generar certificados firmados por el certificado autorizado de la BAD API. Este certificado no está disponible en la primera versión del prototipo.

Si bien la comunicación se puede realizar de forma segura, queda un problema más: que un atacante puede observar la señal que se muestra como forma de feedback y luego intentar reproducirla. La única opción manejable por el servicio BAD API, es que no se envíe la señal a modo de feedback sino que el dispositivo de una señal (luces o similar) al usuario indicándole si está leyendo o no. Por lo tanto los clientes deben tomar los recaudos necesarios para que esto no suceda. Sería posible mostrarle al usuario una especie de tutorial (como el que tiene la aplicación de prueba) que permita al usuario personalizar la lectura, pero al momento del registro de la señal se oculte en la pantalla el feedback.

## 8.2. Entre el servidor del cliente y la BAD API

La comunicación entre ambos se realiza mediante el protocolo HTTPS, donde el cliente debe enviar un *access key* generado de forma segura por la BAD API. Este último aspecto le permite a la BAD API denegar el acceso a cualquier otro servicio que quiera impersonificar al cliente, sin embargo, el guardado de esta clave de forma segura queda en manos del cliente. Se recomienda que se guarde de forma encriptada, o en variables de entorno del servidor teniendo un estricto control de quienes pueden acceder a las mismas.

## 8.3. Entre el dispositivo y la BAD API

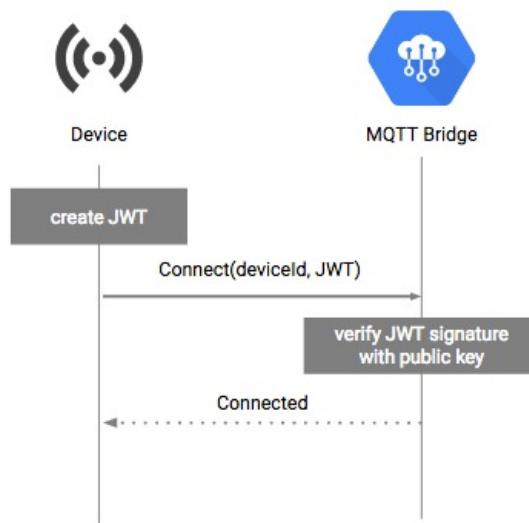


Figura 16: Autenticación de los dispositivos frente a Google Cloud

Si bien se utiliza el protocolo MQTT, que no tiene una transmisión segura, el servicio de Google Cloud le agrega una capa de seguridad al mismo. Los mensajes viajan encriptados al servidor utilizando TLS, previo a una autenticación realizada con criptografía asimétrica (el dispositivo genera un *JWT* firmado con la clave privada, el cual es verificado por Google con la clave pública previamente guardada [Figura 16]).

El registro de la clave pública del dispositivo en los servicios de Google se realizan a nombre de la BAD API y sólo pueden hacerse de forma autenticada [Figura 17]. Tanto el registro como el guardado de la clave pública en el dispositivo son realizados por desarrolladores de la BAD API y no se guarda ningún tipo de copia de la clave privada. La generación del par de claves se realiza con las recomendaciones de seguridad de Google [Good].

Con estos recaudos, sin tener acceso a la clave privada, no es posible que un atacante envíe una señal artificial o altere una señal en tránsito.

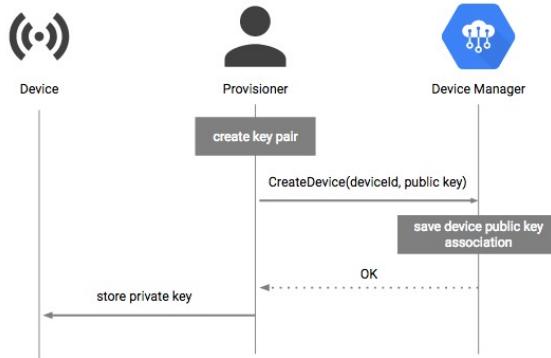


Figura 17: Distribución de claves

#### 8.4. Entre la computadora del usuario y el servidor del cliente.

Esta conexión esta fuera del alcance de la BAD API y debería realizarse al menos utilizando HTTPS. Aquí también toma importancia el hecho de poder eliminar o expirar señales para evitar ataques de replay.

#### 8.5. Guardado del ID de la señal por parte del cliente

Los IDs de las señales son utilizados para enviarlos a la BAD API y que ésta las compare, por este motivo, el cliente debe conservar el valor del mismo sin alterarlo. Normalmente las contraseñas no se almacenan en texto plano ni encriptadas si no se necesita su valor, simplemente se guarda su hash. En este caso, debido a que se necesita el valor para enviarlo, se recomienda que se guarde el ID de forma encriptada.

En este primer prototipo de la BAD API, las señales no tienen ningún tipo de expiración o forma de invalidar las mismas, por lo tanto, es recomendable que los clientes guarden una lista de los IDs que se usaron para logins en el pasado para evitar posibles replay attacks. En un futuro también sería posible agregar a la BAD API una manera de eliminar señales o de agregarle expiración a las mismas para evitar agregarle una carga extra a los clientes.

#### 8.6. Guardado de las claves y la señal en el dispositivo

##### 8.6.1. Normas FIPS

FIPS es un estándar de seguridad para módulos criptográficos que define cuatro niveles. El nivel 1 provee el nivel de seguridad más bajo y tiene requisitos de seguridad básicos como que el módulo utilice algún algoritmo o función criptográfica aprobada. El nivel 2 incluye los requerimientos del nivel 1 y agrega características que muestren evidencia de tampering en el módulo como pueden ser sellos o candados. El tercer nivel también crece sobre el nivel anterior agregando mecanismos de seguridad físicos que puedan detectar y actuar con una alta posibilidad de éxito ante intentos de acceder físicamente al módulo. Esto generalmente consiste en circuitos que borren la información sensible que contiene

el módulo como claves criptográficas. El cuarto y último nivel agrega mecanismos para detectar que el módulo está operando en rangos de voltaje y temperatura fuera de los operativos o garantizar que operar bajo estos rangos no compromete la seguridad del mismo. También requiere un aumento en la probabilidad de detección de accesos provista por el nivel 3.

El prototipo presentado tiene todo lo necesario para al menos conseguir un nivel 1 y no sería difícil agregar lo necesario para llegar al nivel 2. Sin embargo, se buscará que el dispositivo tenga al menos un nivel 3 del estándar FIPS, permitiendo detectar tampering en el dispositivo y borrar la clave privada y certificados almacenados dentro del mismo. Esto evitaría que atacantes puedan utilizar la clave para enviar señales apócrifas a Google para impersonar a los usuarios.

## 9. Lenguaje de programación y bibliotecas utilizadas

El lenguaje de programación utilizado para la realización del *backend* de la BAD API fue Python. La elección del mismo se debió a que es un lenguaje de alto nivel, que permite un desarrollo multiplataforma y principalmente tiene muchas bibliotecas útiles para el desarrollo necesario.

Para desarrollar la aplicación cliente de ejemplo se utilizo Python para el backend y React para el frontend.

Las principales bibliotecas utilizadas fueron:

- Pyserial: herramienta que permite leer de puertos serie, utilizadas para leer la señal del dispositivo.
- Flask: framework para desarrollar aplicaciones web, fue utilizada para la creación de las APIs.
- Orator: ORM que facilita la comunicación con la base de datos.
- Numpy: colección de funciones matemáticas de alto nivel, que facilitan la interpretación de las señales obtenidas.
- Paho-MQTT: cliente del protocolo MQTT que nos permite la comunicación segura entre el dispositivo y la API.

## 10. Metodología de trabajo

Para el desarrollo se utilizó la metodología ágil SCRUM. Para la implementación de dicha metodología utilizamos una herramienta online llamada Trello, que si bien ofrece un panel de la metodología KanBan, lo adaptamos para que siga los lineamientos de SCRUM. Estas adaptaciones incluyen la creación de las columnas Backlog, Doing y Done, que nos permitieron saber en cada momento del trabajo, qué tareas estaba realizando cada integrante de forma paralela y cuales estaban terminadas [Figura 18].

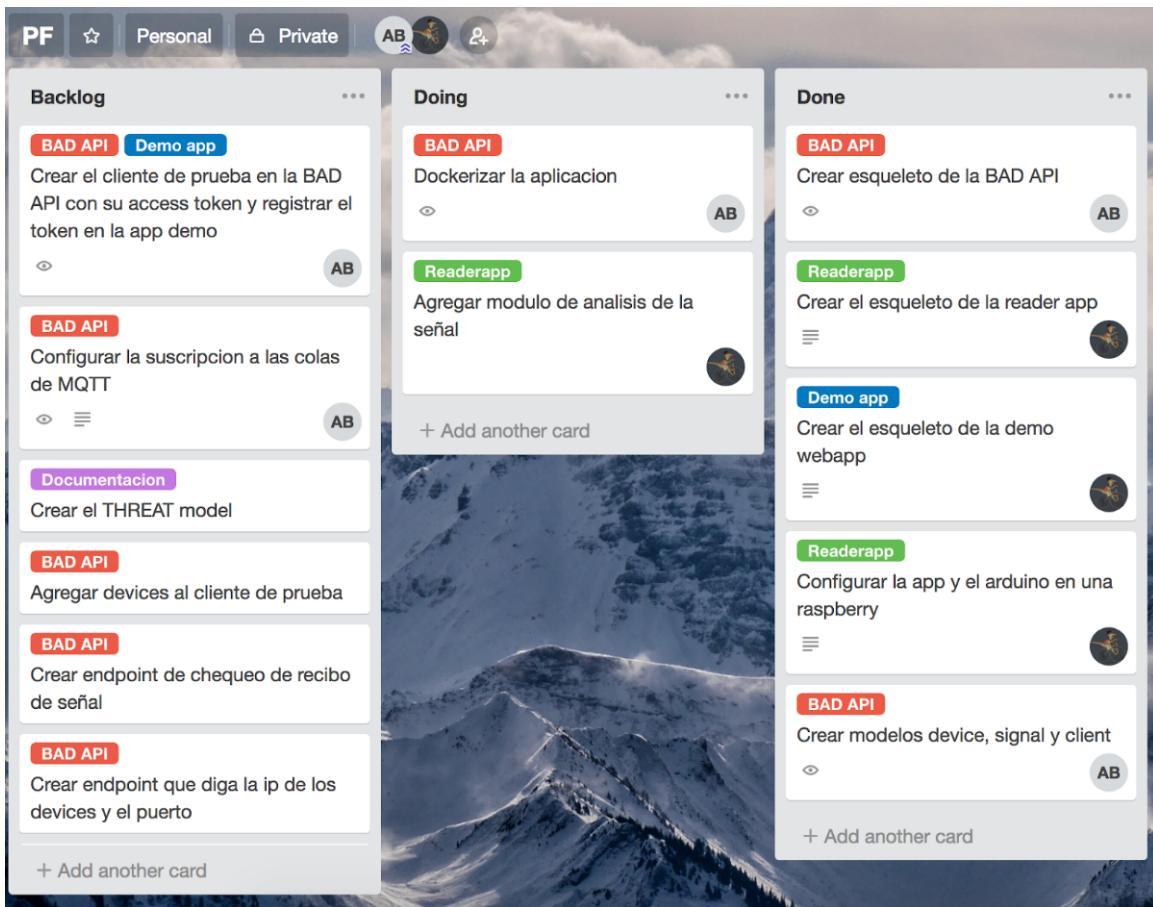


Figura 18: Board de Trello utilizado

Además, utilizamos etiquetas para diferenciar a qué parte del proyecto corresponde de cada tarea. Trello cuenta con filtros para poder ver únicamente las tareas de una determinada etiqueta, que facilita la visión de cada proyecto.

En los inicios de los sprints, definimos qué tareas se iban a realizar y se pasaban de “Backlog” a “Doing” y luego a “Done” una vez que se terminaban. Para esto nos era muy útil tener definidas y estimadas las tareas en “Backlog” para luego poder priorizarlas y elegir cual se realizaba en cada sprint según prioridad y dificultad.

## 11. Implementación de Referencia

Se desarrolló una aplicación de prueba utilizando React en el frontend y Python en el backend para probar las funcionalidades provistas por la BAD API.

Esta aplicación cliente tiene registrados dos dispositivos (Raspberry y laptop) y tiene únicamente las funcionalidades de registro y de login. Los usuarios se registran con un usuario, contraseña y uuid de la señal tomada de la Raspberry.

Dado que la BAD API simplemente compara señales, queda del lado del cliente la decisión de qué hacer con la similitud de las señales y la forma de guardar los UUIDs de las mismas.

Para esta implementación, el cliente guarda en una base de datos PostgreSQL en una tabla “users” el email, la contraseña hasheada con el algoritmo bcrypt y el UUID de la señal obtenida en el registro. Además se optó que el la similitud de las señales debe ser de al menos un 80 % para lograr una autenticación satisfactoria.

## 12. Conclusiones

Si bien no nos fue posible encontrar features características de los individuos para generar un sistema de autenticación, se pudo utilizar perfectamente las señales EMG para el desarrollo de un esquema de autenticación de segundo factor. Aunque la señal se transforma en un código binario, la utilización de un dispositivo de lectura de bioseñales provee una prueba de vida del usuario, aspecto que carecen los secretos compartidos.

Creemos que la clave para la masificación de este tipo de tecnologías es la implementación de una interfaz genérica por encima de los propios drivers de los dispositivos de lectura de señales, en nuestro caso fue una interfaz HTTP pero podría ser otra. Esto permite poder adaptar todo tipo de aplicaciones a la utilización de dichas interfaces de manera sencilla, evitando los tipos problemas que se tuvieron con el dispositivo de la marca TrueSense y, además, permite mejorar el dispositivo de captura de la señal sin necesidad de alterar las aplicaciones que lo utilizan.

La tecnología está en un grado de madurez y costo suficientemente bajo como para que se comiencen a utilizar las bioseñales como forma de autenticación de forma obligatoria en servicios que manejan información sensible de los usuarios como los bancarios.

Uno de los problemas más importantes que se presentan cuando se utilizan sistemas de autenticación mediante información intrínseca de las personas, como huellas digitales o iris, es que ante el caso de que se filtre esa información (por un ataque, por ejemplo), los usuarios no pueden cambiarla. Por eso, lo más interesante de las bioseñales es que requieren la participación activa de los usuarios que otorgan una prueba de vida y aunque la información se extravíe, es más complicado impersonificar a otro usuario ya que es posible agregar varias formas de protección a los dispositivos.

## 13. Disclaimer

Tanto la API como el prototipo no pretenden ser una solución completa y probada sino un punto de partida para eventualmente ofrecer una aplicación comercial. Para evaluar

la robustez del sistema es necesario que haya un escrutinio o *penetration testing* realizado por expertos de seguridad. Finalmente para la comercialización es fundamental conseguir el certificado FIPS de al menos nivel 3 y los certificados para TLS.

## 14. Trabajo futuro

Como trabajo a futuro se planea agregar nuevos dispositivos que utilicen otras funciones biométricas de EMG como EEG (Muse, OpenBCI) para lo cual necesitan de conexión bluetooth LE. Dentro de las posibles mejoras del desarrollo actual se puede utilizar algún dispositivo de lectura de señales de menor tamaño y/o que pueda obtener señales con mayor velocidad y/o fidelidad. Estos últimos puntos son útiles para aumentar la aceptación psicológica de los usuarios.

Una aplicación relacionada sería la utilización de esta tecnología para identificar a la persona en base a las señales de EMG en el momento en que realiza una firma sobre un papel, su firma de puño y letra. Este esquema obtendría un fingerprint de las señales de EMG similares a lo realizado en este trabajo, con el agregado del componente activo que agregue un eje más de verificación y que permite la actualización de la información complementaria en el caso de que la misma sea comprometida. Además generaría un beneficio en cuánto a la aceptación de la misma ya que la firma de puño y letra es un mecanismo natural para proveer de una autenticidad a documentos legales y burocráticos. En caso de funcionar, podría transformarse en una forma muy útil de firmar pagos electrónicos.

En el trabajo realizado en la universidad de La Plata [Hab16] se realizó un estudio específico de los biopotenciales de EMG en una aplicación para personas con discapacidades motoras. En el mismo se implementa un prototipo de dispositivo de ayuda tipo switch que emite pulsos binarios generados a partir de señales EMG. Un posible trabajo futuro podría ser realizar una interfaz genérica que permita comunicar este tipo de dispositivos con aplicaciones web o de escritorio de forma tal que sea más fácil la incorporación de nuevos dispositivos.

Además de los usos actuales de EMG y EEG, otros usos innovadores de estas señales podrían ser en la industria automotriz (por ejemplo para que los autos solo puedan ser abiertos y encendidos por ciertas personas, para activar el manejo autónomo o incluso para manejar) o el manejo de drones.

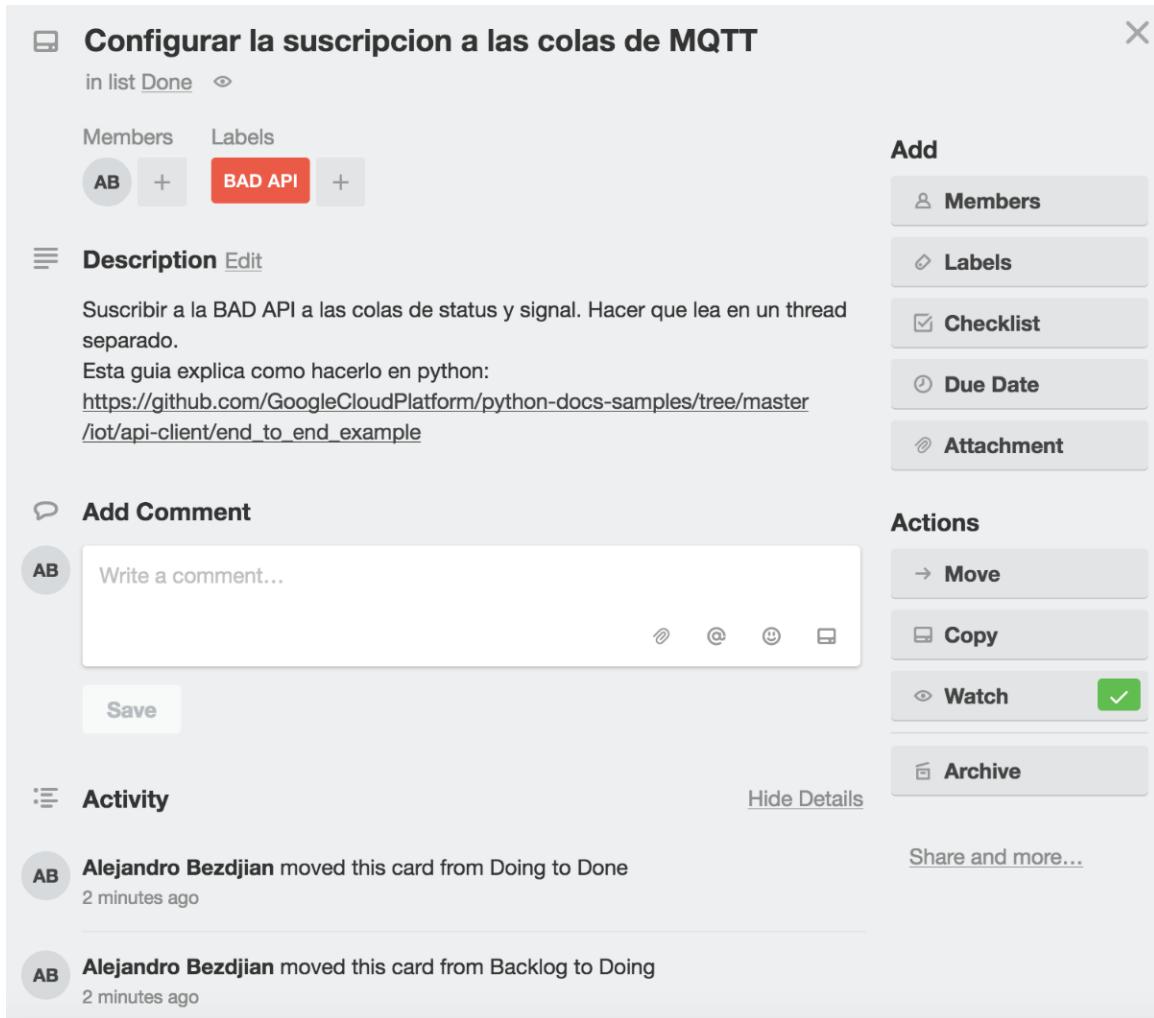


Figura 19: Información de una “card” de Trello

# BAD Client demo

Si es la primera vez que entras, podés realizar el [tutorial](#)



Figura 20: Se muestra la señal utilizada para el registro de un usuario de prueba

# BAD Client demo

Logueado correctamente. Porcentaje de similitud: %85

Bienvenido test@example.com!

Figura 21: La similitud de las señales supera el umbral elegido por el cliente y se autentica

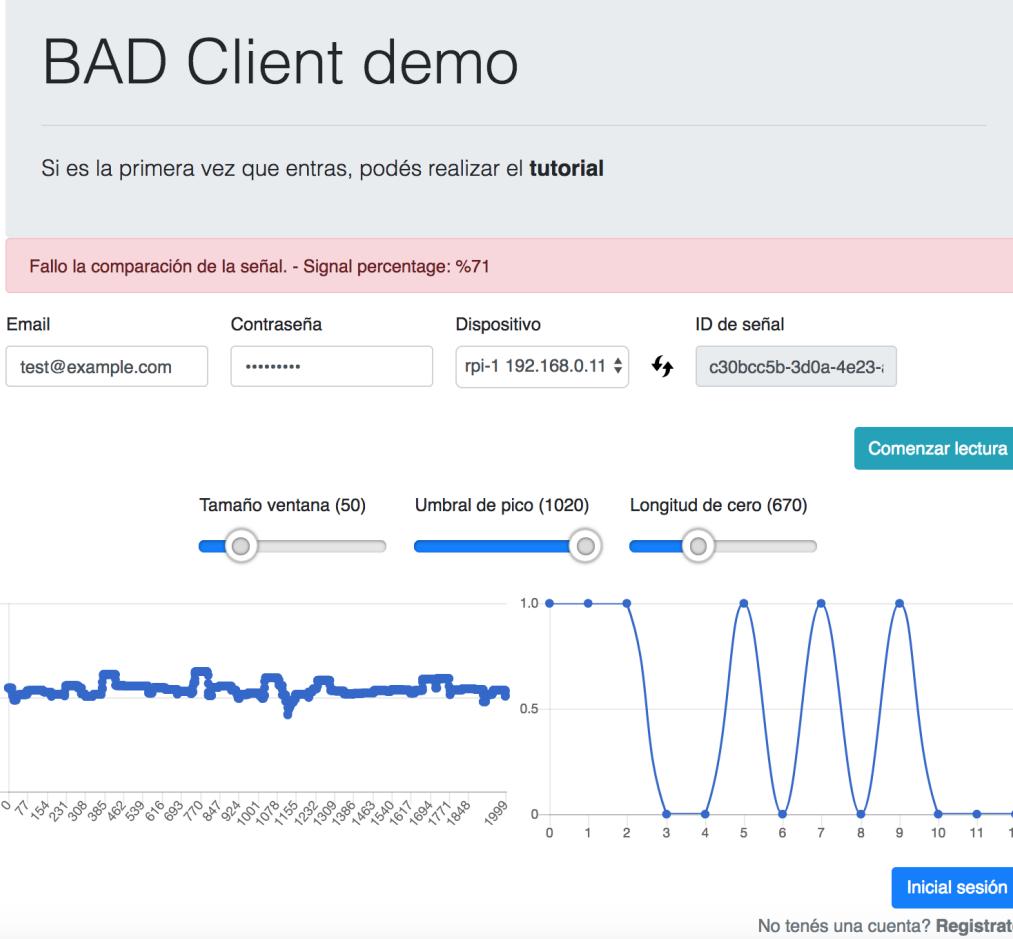


Figura 22: La autenticación falla porque el porcentaje de similitud de las señales no supera el umbral elegido por el cliente de 80 %

## Bibliografía

- [MBI06] F. Mohd-Yasin M.B.I. Raez M.S. Hussain. “Techniques of EMG signal analysis: detection, processing, classification and applications”. En: (2006). DOI: <https://dx.doi.org/10.1251%2Fbpo115>. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1455479/>. (accessed: 30.07.2018).
- [Sae07] J.A. Chambers Saeid Sanei. *EEG Signal Processing*. John Wiley & Sons Ltd, 2007. ISBN: 9780470025819.
- [Ang09] Pornchai Phukpattaranont Angkoon Phinyomark Chusak Limsakul. “Feature Extraction for Robust EMG Pattern Recognition”. En: *A Journal of computing* 1 (2009). ISSN: 2151-9617.
- [Bre12] Eli Bressert. *SciPy and NumPy*. O'Reilly Media, 2012.
- [Rub13] Mohd Alauddin Bin Mohd Ali Rubana H. Chowdhury Mamun B. I. Reaz. “Surface Electromyography Signal Processing and Classification Techniques”. En: (2013). DOI: 10.3390/s130912431. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3821366>. (accessed: 30.07.2018).
- [Abo15] Ahmad Taher Azar Aboul Ella Hassanien. *Brain-Computer Interfaces Current Trends and Applications*. Springer, 2015.
- [Ail15] Alcimar Barbosa Soares Ailton Luiz Dias Siqueira Júnior. “A novel method for EMG decomposition based on matched filters”. En: *Res. Biomed. Eng.* 31.1 (2015). URL: [http://www.scielo.br/scielo.php?script=sci\\_arttext%5C&pid=S2446-47402015000100044](http://www.scielo.br/scielo.php?script=sci_arttext%5C&pid=S2446-47402015000100044). (accessed: 30.07.2018).
- [Bel+15] Noureddine Belgacem y col. “A novel biometric authentication approach using ECG and EMG signals”. En: *Journal of Medical Engineering & Technology* 39.4 (2015), págs. 226-238. DOI: 10.3109/03091902.2015.1021429. eprint: <https://doi.org/10.3109/03091902.2015.1021429>. URL: <https://doi.org/10.3109/03091902.2015.1021429>. (accessed: 30.07.2018).
- [Cha15] Preeti Singh Chamandeep Kaur. En: (2015). DOI: <https://dx.doi.org/10.1155\%2F2015\%2F614723>. URL: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4684838/>. (accessed: 30.07.2018).
- [Oli15] Travis E. Oliphant. *Guide to NumPy*. CreateSpace Independent Publishing Platform, 2015.
- [Hab16] Marcelo Haberman. “Procesamiento de señales aplicado a dispositivos de ayuda para personas con discapacidades motoras”. Tesis doct. 2016. URL: [http://sedici.unlp.edu.ar/bitstream/handle/10915/53022/Documento\\_completo.pdf-PDFA.pdf](http://sedici.unlp.edu.ar/bitstream/handle/10915/53022/Documento_completo.pdf-PDFA.pdf). (accessed: 30.07.2018).
- [Lan16] Hans Petter Langtangen. *A Primer on Scientific Programming with Python*. Springer, 2016.
- [Moo16] Sungho Jo Moonwon Yu Netiwit Kaongoen. “P300-BCI-Based Authentication System”. En: (2016). DOI: 10.1109/IWW-BCI.2016.7457443.
- [Tan16] Karan Veer Tanu Sharma. “Comparative study of wavelet denoising in myoelectric control applications”. En: *Journal of Medical Engineering and Technology* (2016). DOI: 10.3109/03091902.2016.1139200.

- [Yan16] Chao Zhang Yang Liu Xuetao Feng. *Electrocardiogram (ECG) authentication method and apparatus*. 2016. URL: <http://www.freepatentsonline.com/20170188971.pdf>. (accessed: 30.07.2018).
- [Cha17] Maurice Charbit. *Digital Signal Processing with Python Programming*. Wiley, 2017.
- [Esp17] Jose Antonio Martín Esparza. *Los datos biométricos son datos personales*. 2017. URL: <https://www.bloginnova.com/2017/04/biometria.html>. (accessed: 30.07.2018).
- [KP17] Jin Su Kim y Sung Bum Pan. “A Study on EMG-based Biometrics”. En: *Journal of Internet Services and Information Security (JISIS)* (2017).
- [Gooa] Google. URL: <https://cloud.google.com/iot-core>. (accessed: 30.07.2018).
- [Goob] Google. URL: <https://cloud.google.com/appengine>. (accessed: 30.07.2018).
- [Gooc] Google. URL: <https://cloud.google.com/sql>. (accessed: 30.07.2018).
- [Good] Google. URL: <https://cloud.google.com/iot/docs/concepts/device-security>. (accessed: 30.07.2018).
- [Hun] Troy Hunt. URL: <https://haveibeenpwned.com>. (accessed: 30.07.2018).
- [OPI] OPIInovations. *ReLax Operations and APP Download*. URL: <http://www.op-innovations.com/en/ReLaxAPP>. (accessed: 30.07.2018).
- [Sha] Larry Koved Shari Trewin Cal Swart. “Biometric Authentication on a Mobile Device: A Study of User Effort, Error and Task Disruption”. En: (). URL: <https://researcher.watson.ibm.com/researcher/files/us-kapil/ACSAC12.pdf>. (accessed: 30.07.2018).
- [Vee] Karan Veer. “Experimental Study and Characterization of SEMG Signals for Upper Limbs”. En: (). DOI: [dx.doi.org/10.1142/S0219477515500285](https://dx.doi.org/10.1142/S0219477515500285).