CAPÍTULO 5. EVALUACIÓN Y VALIDACIÓN DE LA PROPUESTA

En esta sección se detallan los experimentos realizados para poder evaluar y validar la propuesta. En primer lugar, se evalúa el flujo de trabajo propuesto en el Capítulo 3. Finalmente, se evalúa el método propuesto en el Capítulo 4 para la combinación de información de comportamientos.

5.1. Evaluación del flujo de trabajo

5.1.1. Desarrollo del entorno de trabajo y conjunto de datos UEBA

Con el objetivo de poder evaluar el flujo de trabajo propuesto, se ha desarrollado un entorno de trabajo totalmente funcional que implementa la gestión de identidades federada. Para ello, se ha desarrollado una aplicación de chat, cuyo rol es el de RP, y un IdP cuya función es la de gestionar las identidades digitales necesarias para realizar los experimentos.

El chat se ha desarrollado utilizando la aplicación de código abierto *Lets-chat* [142], el cual se encuentra bajo la licencia MIT. Esta aplicación se ha desarrollado en *Node.js* [143] y utiliza una base de datos *MongoDB* [144] para almacenar toda la información necesaria. La aplicación es compatible con la mayoría de navegadores web del mercado y es multidipositivo. En el presente trabajo de tesis, esta aplicación ha sido ejecutada en un ordenador portátil MSI GF62 8RD-256XES equipado con un procesador Intel Core i7 8750H (2.2 GHz, 9MB). Además, tiene una memoria RAM de 16 GB DDRIV (2666 MHz).

Por otro lado, se ha desarrollado un componente que se integra dentro del chat con el objetivo de recopilar la información de comportamiento. Este componente, desarrollado en *Javascript* y basado en el trabajo [145], se integra en el cliente web donde se ejecuta la aplicación. Su funcionalidad es la de recoger la información relacionada con los eventos de pulsación de teclas en el teclado y de movimientos de ratón y almacenarla. Los eventos relacionados con el teclado son recopilados cada vez que suceden, sin embargo, los eventos de ratón se han de recoger por *pooling* , es decir, cada cierto tiempo. En este caso concreto los eventos de *pooling* se han recopilado cada 200 milisegundos debido a los recursos de memoria y computación disponibles.

Finalmente, el IdP se ha desarrollado utilizando el *framework* OpenAM [146]. De esta forma, se ha hecho uso del *Web application Resource* (WAR) para

instalar la versión 13.5. Una vez instalado, y teniendo en cuenta los requisitos del presente trabajo, se ha implementado un cliente OAuth 2.0/OpenID Connect. Este cliente proporciona de forma sencilla todos los mecanismos para poder realizar los flujos de autorización y autenticación disponibles en dichas especificaciones.

Este desarrollo, ha dado lugar a el conjunto de datos denominado *Keystroke and Mouse Dynamics for UEBA Dataset*, el cual ha sido recopilado para el presente trabajo de tesis doctoral y está disponible públicamente para la comunidad científica [147]. Para recopilar el conjunto de datos, un grupo de once investigadores de la Universidad Rey Juan Carlos heterogéneos en cuanto a edad y género, han hecho uso de la aplicación de chat. Todos los participantes, mayores de edad, han recibido información sobre la finalidad del experimento y han participado en él de manera completamente voluntaria, sin recibir ningún tipo de presión y proporcionando para ello su consentimiento explícito. A todos se les ha informado de los mínimos riesgos que corrían, ya que solo se recogían datos relacionados con el uso explícito de la aplicación de chat, que no permiten identificarles y que además no se asociaban a sus datos personales. De este modo, se han recopilado dos bases de datos denominadas *EVTRACKINFO* y *EVTRACKTRACK*. Estas bases de datos mantienen una relación 1:N entre ellas respectivamente. *EVTRACKINFO* contiene información de sesión, es decir, almacena los atributos estáticos como, por ejemplo, el usuario autenticado, agente de usuario y la resolución de la pantalla asociado a cada sesión (ver Sección 3.3.1). Por otro lado, *EVTRACKTRACK* contiene la información asociada a las dinámicas de comportamiento de cada sesión. Esto es, los eventos derivados de las pulsaciones de teclado y de ratón.

En total, 347 y 142691 registros han sido recopilados para *EVTRACKINFO* y *EVTRACK-TRACK* respectivamente. Del total de las dinámicas de comportamiento, 113471 pertenecen a dinámicas de teclado, mientras que 29220 pertenecen a dinámicas de ratón. Para cada usuario, se han obtenido un total de 28524 ± 18541 registros (media ± desviación típica).

El entorno de trabajo desarrollado y el conjunto de datos obtenido se han utilizado para evaluar y validar todas las tareas que componen el flujo de trabajo propuesto. De aquí en adelante, se detalla la implementación de cada una de estas tareas.

5.1.2. Selección de la huella digital en el conjunto de datos UEBA

En el caso del chat desarrollado, se ha seleccionado una RP con un perfil de seguridad medio (ver Sección 3.3.1). Hay que destacar, que de acuerdo con los bajos recursos computacionales del servidor donde se ejecuta

la aplicación y a la gran cantidad de usuarios que pueden interactuar con la aplicación de forma simultánea, la huella digital debe ser lo más simple posible con el objetivo de no saturar el sistema, pero lo suficientemente discriminatoria para poder diferenciar entre los distintos comportamientos de los usuarios.

De esta forma, el agente de usuario y la resolución de pantalla se han seleccionado como atributos estáticos, mientras que las dinámicas de teclado y las dinámicas de ratón se han seleccionado como atributos dinámicos. Cabe destacar que, debido a la naturaleza de la aplicación de chat, se espera obtener una recopilación masiva de datos, ya que el teclado y el ratón van a ser utilizados asiduamente por los usuarios.

5.1.3. Generación de la huella digital en el conjunto de datos UEBA

El objetivo de esta tarea es generar una huella digital para cada usuario con la suficiente singularidad como para ser distintiva entre usuarios. De esta forma, los atributos estáticos y los atributos dinámicos son recopilados haciendo uso del componente específicamente desarrollado para ello. Esta información está recogida en bruto y, por tanto, se trata para poder extraer conocimiento y generar características significativas que puedan alimentar a los modelos de aprendizaje máquina. En el caso de los atributos estáticos la información en bruto es transformada directamente a variables categóricas. En el caso de los atributos dinámicos se distinguen dos posibles casuísticas: las dinámicas de teclado y las dinámicas de ratón.

En el caso de las dinámicas de teclado, cada pulsación de una tecla, es decir, cada interacción, genera dos tipos de eventos principalmente (debido a la tecnología seleccionada para su implementación): *keydown* y *keyup*. El evento de *Keydown* contiene la marca de tiempo sobre cuándo se ha inicializado la pulsación, mientras que el evento *keyup* contiene la marca de tiempo de cuando la pulsación ha concluido, es decir, cuando el usuario ha dejado de mantener la tecla pulsada. Estas interacciones se agrupan en ventanas temporales de dos pulsaciones formando digrafos. De esta forma, cada digrafo contiene cuatro marcas de tiempo (para cada pulsación un evento de *keyupX* y otro de *KeydownX*, donde X representa el número de interacción). De esta forma, se calculan seis características o variables para cada digrafo [105]:

- H1 (*keyup1-Keydown1*): tiempo transcurrido para la primera interacción.
- H2 (*Keyup2-Keydown2*): tiempo transcurrido para la segunda interacción.
- HP (*Keydown2-Keyup1*): tiempo transcurrido desde que termina la primera interacción hasta que se inicia la segunda interacción.

- PP (*Keydown2-Keydown1*): tiempo transcurrido desde que se inicia la primera interacción hasta que se inicia la segunda interacción.

- HH (*Keyup2-Keyup1*): tiempo transcurrido desde que acaba la primera interacción hasta que acaba la segunda interacción.

- PH (*Keyup2-Keydown1*): tiempo transcurrido desde que se inicia la primera interacción hasta que acaba la segunda interacción.

Además, se almacena información sobre la propia tecla pulsada, esto es, se almacenan los caracteres específicos pulsados para cada digrafo. Estos caracteres se categorizan teniendo en cuenta una división física del teclado, seleccionando las teclas que se encuentran a la izquierda del teclado y aquellas a la derecha en otro grupo. También se incluye la posición en el eje vertical, arriba y abajo para cada uno de los grupos anteriormente mencionados. De esta forma, esta categoría puede tomar cuatro valores distintos.

En el caso de las dinámicas del ratón, se han extraído cinco características. Al igual que en el caso del teclado, los eventos del ratón se han agrupado formando digrafos. Para cada digrafo se ha calculado: el ángulo, la distancia, la velocidad, el tiempo total transcurrido y el tipo de evento de *Javascript*.

Utilizando las características recopiladas, se genera una huella digital para cada usuario. Esta huella contiene tres componentes: el componente de los atributos estáticos, el componente de las dinámicas de teclado y el componente de las dinámicas de ratón. La información de cada uno de los componentes está almacenada en forma de vector.

5.1.4. Modelado y evaluación

Los procesos de modelado y evaluación tienen como objetivo construir los modelos de aprendizaje máquina, basados en UEBA, que clasifican las dinámicas de comportamiento. Estos procesos se han considerado conjuntamente en esta sección, pues se van a realizar de forma simultánea, avanzando y retrocediendo hasta llegar a unos modelos con alta capacidad de generalización, robustos y con eficacia alta.

Los experimentos aquí detallados se han centrado en desarrollar un modelo de aprendizaje máquina capaz de comparar y evaluar las huellas digitales generadas en los pasos anteriores. En primer lugar, los datos recopilados se han separado en tres conjuntos: conjunto de entrenamiento, test y validación. El conjunto de entrenamiento contiene el 70 % de los datos de cada usuario específico (muestras genuinas). Los conjuntos de test y de validación contienen, respectivamente, un 15 % de los datos restantes. Los conjuntos de

test y validación son completados utilizando muestras atípicas para simular comportamientos anómalos, es decir, comportamientos que pueden suponer una brecha de seguridad y por tanto se quieren detectar (muestras de impostores). Para llevarlo a cabo, se seleccionan de forma aleatoria tantas muestras del resto de usuarios como tenga el propio usuario para cada uno de los conjuntos. En conclusión, el conjunto de entrenamiento solo contiene muestras genuinas y se utiliza para entrenar los modelos de detección de anomalías pertinentes. Por otro lado, los conjuntos de test y validación contienen muestras tanto genuinas, como anómalas, y se utilizan para evaluar la eficacia de los modelos propuestos y determinar si son robustos y capaces de generalizar correctamente.

En cuanto a la comparación de huellas digitales, cada componente (atributos estáticos, dinámicas de teclado y dinámicas de ratón) se modela de forma independiente. De esta forma, los componentes pueden ser activados o desactivados para poder evaluar la comparación de huellas digitales de forma independiente para cada uno de ellos.

En primer lugar se ha implementado un NB para comparar los atributos estáticos. Este clasificador asume que cada variable es totalmente independiente y, por lo tanto, cada variable determina la probabilidad de pertenecer a la clase genuina o a la impostora [148]. Debido al bajo número de participantes en el conjunto de datos UEBA, los resultados obtenidos muestran una clasificación perfecta. Esto quiere decir que simplemente considerando el agente de usuario y la resolución de pantalla, este simple clasificador es capaz de determinar si un usuario es genuino o impostor de forma exacta. En el caso de que el número de usuarios aumentase, el número de atributos estáticos que habría que considerar sería mayor y muy probablemente no se conseguirá esta clasificación perfecta. Además, estos atributos estáticos son fáciles de manipular y falsear (para los atributos dinámicos la complejidad de manipulación o falseo es mayor o incluso imposible). Por otro lado, ciertos ataques pueden tomar el control de la máquina del usuario genuino y por lo tanto hacer que estos atributos sean idénticos a los esperados e imposibles de detectar por el modelo. Es por esto que en caso de que el número de usuarios aumente, se recomienda aumentar el nivel de seguridad de las RPs (actualmente se ha seleccionado un nivel medio de seguridad).

En el caso de las dinámicas de teclado, los digrafos generados se agrupan en n-gramas [149]. De esta forma se concatenan los digrafos formando vectores de longitud dos, tres y seis. Sobre estos vectores se utiliza la distancia de Manhattan [150] para comparar las dinámicas de comportamiento asociadas al teclado [85].

El conjunto de entrenamiento para cada usuario se utiliza para calcular la media de cada característica recopilada. Posteriormente se extraen los

vectores de características para el conjunto de test. De esta forma se utiliza la distancia de Manhattan para comparar estos vectores con la media calculada en el conjunto de entrenamiento. De este modo se obtiene la similitud entre estas muestras y lo esperado. Con estas similitudes calculadas en el conjunto de test se puede fijar un umbral de decisión utilizando el EER. En caso de que la similitud entre las muestras sea menor al umbral determinado se clasifica como genuina, mientras que en caso de que supere el umbral se clasifica como impostor.

Con el umbral ya fijado, se procede a evaluar el conjunto de validación. Los datos almacenados en este conjunto no han sido nunca antes vistos por el modelo y, por lo tanto, permiten evaluar el modelo de forma justa y analizar la capacidad de generalización del modelo obtenido. Los resultados obtenidos para este modelo son bastante pobres, obteniendo un EER de 0,511 ± 0,124 (media ± desviación típica). Estos resultados se obtienen porque los datos recopilados no tienen una frecuencia o distribución predeterminada, a diferencia de otros trabajos del estado del arte que sí asumen un patrón determinado (usuarios introduciendo la misma contraseña un número determinado de veces). Es por esto que se deben incluir ciertas mejoras en el modelo para obtener unos resultados óptimos.

Siguiendo esta línea se ha realizado un proceso de exploración de datos más exhaustivo, permitiendo determinar que los vectores que comienzan con la misma pulsación, tienden a ser más similares entre sí que los que comienzan con una pulsación distinta (p. ej. los vectores que comienzan por la pulsación del carácter 'a' son más similares entre sí, a los que comienzan por el carácter 'b' y viceversa). De esta forma, se han agrupado los vectores en función de su primera pulsación. En este caso, también se ha observado que el vector medias que se utiliza en el conjunto de entrenamiento no es lo suficientemente discriminatorio debido a la alta dispersión de los datos de cada usuario. Es por esto que se ha decidido sustituir este vector medias por un modelo de KNN utilizando la misma distancia de Manhattan. De esta forma, cada muestra no se va a comparar contra el vector media de características, sino contra la etiqueta conocida que posean los n-vecinos más cercanos. Al igual que para el modelo anterior, se utiliza el EER para fijar un umbral óptimo de clasificación.

Los resultados obtenidos se pueden observar en la Tabla 5.1. Estos resultados confirman que el modelo mejorado representa de forma óptima el comportamiento de los usuarios y por consiguiente clasifica correctamente un número elevado de muestras, es decir, obtiene valores de EER FAR y FRR menores.

El modelo obtenido compara cada vector de características de forma totalmente independiente. Sin embargo, en un entorno real, estos vectores de características se van generando a lo largo del tiempo y de forma ordenada

Tabla 5.1. Resultados del modelo de clasificación KNN utilizando la distancia de Manhattan para las dinámicas de teclado agrupando por tecla pulsada: FAR, FRR y EER (desviación típica)

	Test			Validación	
N-grama	FAR	FRR	EER	FAR	FRR
2	0,257	0,265	0,296 (0,099)	0,321	0,304
3	0,284	0,294	0,308 (0,099)	0,317	0,309
6	0,279	0,294	0,347 (0,141)	0,267	0,299

a medida que los usuarios van interaccionando con la aplicación. Esto se traduce, en que las distancias obtenidas pueden también ordenarse y analizarse a lo largo del tiempo y no compararse una a una como se ha hecho hasta ahora. Para realizar esto, se propone utilizar un *buffer* temporal. En este caso, las distancias entre vectores se utilizan para llenar el *buffer*. De esta manera, si la distancia obtenida para un vector concreto supera el umbral determinado, el *buffer* se llenará. En caso contrario, es decir, si la distancia es menor que el umbral, el *buffer* disminuirá. El valor a aumentar o disminuir en el *buffer* es la distancia a dicho umbral. De esta forma, se determina un nuevo umbral de clasificación para este *buffer* basándose, al igual que para los modelos previos, en el menor EER.

A continuación se utilizan los conjuntos de test y validación para evaluar el modelo. En esta ocasión, los datos pertenecientes a usuarios impostores se concatenan al final de las muestras genuinas. Los resultados para este experimento se encuentran en la Tabla 5.2. Tal y como se puede observar, incluir la información temporal hace que el modelo aumente considerablemente su eficacia a la hora de clasificar las muestras de test correctamente. Además, el modelo obtenido es robusto, pues también obtiene resultados

Tabla 5.2. Resultados del modelo de clasificación KNN utilizando la distancia de Manhattan y el *buffer* temporal para las dinámicas de teclado: FAR, FRR y EER (desviación típica). Modelo de referencia

	Test			Validación	
N-grama	FAR	FRR	EER	FAR	FRR
2	0,125	0,128	0,133 (0,107)	0,012	0,156
3	0,087	0,091	0,099 (0,126)	0,050	0,082
6	0,130	0,140	0,175 (0,156)	0,068	0,144

semejantes y óptimos para el conjunto de validación. Sin embargo, este modelo contiene un sesgo pues se está asumiendo que los comportamientos anómalos suceden siempre al final de las muestras genuinas, algo que en un escenario real no es así necesariamente. En un escenario real las anomalías de comportamiento pueden suceder en cualquier instante temporal pues no se puede saber a priori cuando un usuario va a sufrir un ataque ni la duración del mismo. Es por esto que el modelo generado, a pesar de no ser una buena solución para un entorno real, sirve para corroborar que, si se considera la suficiente información de comportamientos, se pueden obtener resultados óptimos a la hora de clasificar las muestras. Es decir, sirve para generar un modelo de referencia cuyos resultados son los mejores que se pueden obtener siguiendo los pasos aquí descritos.

Con el objetivo de definir un experimento más realista y semejante a un entorno real, los vectores de características se agrupan en sesiones utilizando ventanas temporales. De esta forma, los vectores de características se agrupan en sesiones de longitud 5, 10 y 20. Por ejemplo, para una longitud de sesión de 10, 10 vectores genuinos se concatenan con 10 vectores impostores. Este vector resultante va a formar una sesión independiente. Los resultados obtenidos para esta situación más realista se pueden encontrar en la Tabla 5.3 y la Tabla 5.4. Los mejores resultados se obtienen utilizando n-gramas de longitud 2 y 3. Los resultados obtenidos son peores que los obtenidos para el modelo que se toma como referencia en el paso anterior, sin embargo, se llega a una clasificación mejor a medida que el tamaño de sesión aumenta. Esto corrobora que, cuanta más información es considerada, mejores resultados se obtienen.

La Figura 5.1 ilustra un ejemplo de los valores del *buffer* para un usuario concreto. La línea azul representa el valor del *buffer* a lo largo del tiempo. La línea negra horizontal representa el umbral de decisión. La línea verde representa que una predicción se ha realizado correctamente, tanto en el caso de clasificar una muestra genuina como en el de clasificar una muestra de un impostor. La línea roja representa una predicción errónea.

Finalmente, los mismos pasos se han seguido para el caso de las dinámicas de ratón. En primer lugar, se han utilizado n-gramas de longitud 1,2,3 y 6. Posteriormente, se ha considerado la distancia de Manhattan para comparar de forma independiente cada vector contra el vector medias. Al igual que para el caso del teclado, los resultados obtenidos son bastante pobres, obteniendo un EER de 0,496 ±0,160 para la mejor selección del parámetro (longitud de n-grama 6). En este caso concreto, para mejorar el modelo inicial, se ha optado por utilizar OC-SVM. Los resultados obtenidos se pueden observar en la Tabla 5.5. Estos resultados no se pueden comparar con los

Figura 5.1. Valores del *buffer* utilizando las dinámicas de teclado para un usuario concreto

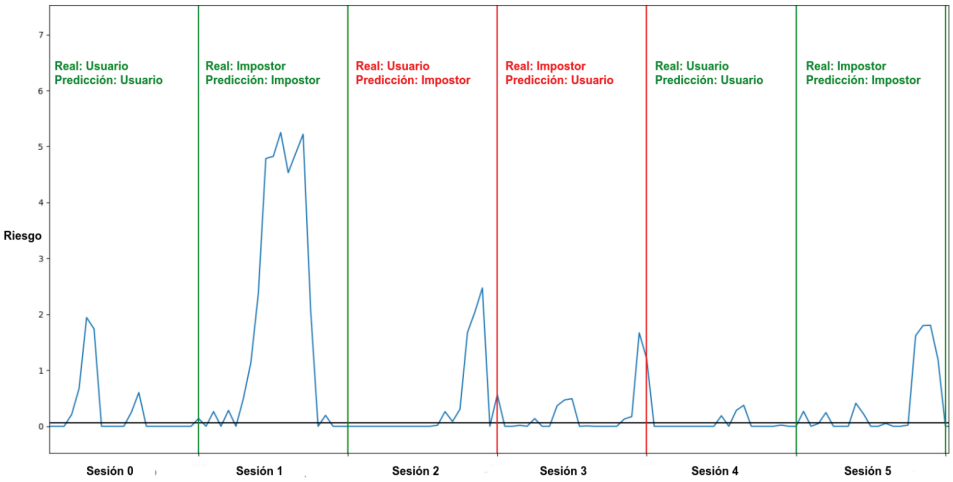


Tabla 5.3. Resultados del modelo de clasificación KNN utilizando la distancia de Manhattan y el *buffer* temporal para las dinámicas de teclado divididas en sesiones utilizando en el conjunto de test: FAR, FRR y EER (desviación típica)

Tamaño de sesión	N-grama	FAR	FRR	EER
5	2	0,221	0,213	0,238 (0,172)
5	3	0,254	0,242	0,205 (0,136)
5	6	0,235	0,197	0,247 (0,195)
10	2	0,169	0,162	0,279 (0,120)
10	3	0,185	0,194	0,188 (0,157)
10	6	0,221	0,228	0,206 (0,206)
20	2	0,141	0,162	0,128 (0,140)
20	3	0,172	0,140	0,145 (0,189)
20	6	0,153	0,172	0,174 (0,208)

Tabla 5.4. Resultados del modelo de clasificación KNN utilizando la distancia de Manhattan y el *buffer* temporal para las dinámicas de teclado divididas en sesiones utilizando en el conjunto de validación: FAR y FRR

Tamaño de sesión	N-grama	FAR	FRR
5	2	0,199	0,220
5	3	0,213	0,213
5	6	0,262	0,211
10	2	0,170	0,216
10	3	0,170	0,170
10	6	0,155	0,196
20	2	0,126	0,188
20	3	0,161	0,110
20	6	0,155	0,180

Tabla 5.5. Resultados del modelo de clasificación OC-SVM para las dinámicas de ratón de forma independiente: FAR, FRR y EER (desviación típica)

	Test			Validación	
N-grama	FAR	FRR	EER	FAR	FRR
1	0,447	0,448	0,455 (0,028)	0,440	0,441
2	0,442	0,443	0,427 (0,058)	0,444	0,447
3	0,440	0,440	0,432 (0,060)	0,442	0,445
6	0,447	0,419	0,407 (0,081)	0,476	0,468

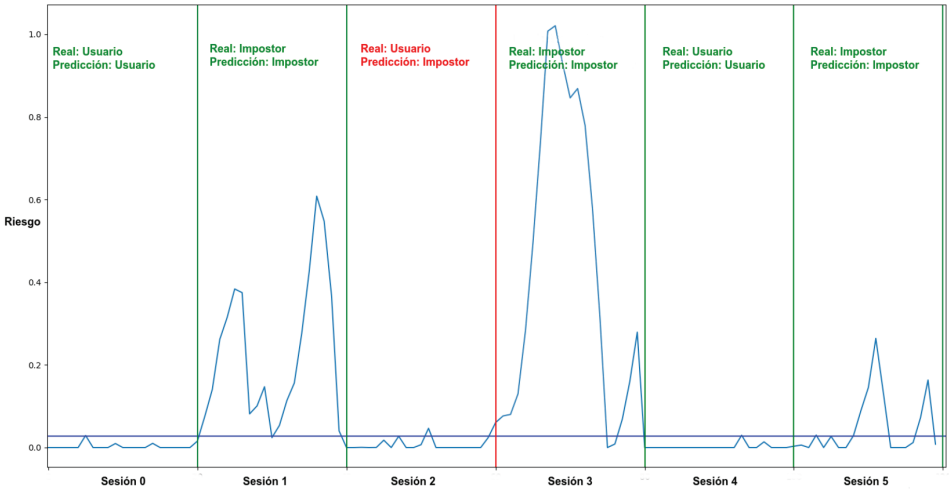
obtenidos para las dinámicas de teclado pues pertenecen a dos fuentes de datos totalmente distintas.

Siguiendo las líneas definidas para el caso de las dinámicas de teclado, se procede a utilizar un *buffer* temporal. En esta primera aproximación, para los conjuntos de test y de validación, se concatenan todas las muestras de impostores al final de las muestras genuinas. Los resultados obtenidos se pueden observar en la Tabla 5.6. Estos son los resultados pertenecientes al modelo referencia.

Tabla 5.6. Resultados del modelo de clasificación OC-SVM utilizando el *buffer* temporal para las dinámicas de ratón: FAR, FRR y EER (desviación típica). Modelo de referencia

	Test			Validación	
N-grama	**FAR**	**FRR**	**EER**	**FAR**	**FRR**
1	0,072	0,108	0,151 (0,083)	0,115	0,093
2	0,205	0,214	0,249 (0,162)	0,168	0,088
3	0,399	0,399	0,409 (0,081)	0,306	0,173
6	0,415	0,447	0,399 (0,088)	0,460	0,479

Figura 5.2. Valores del *buffer* utilizando las dinámicas de ratón para un usuario concreto



Con el objetivo de contemplar un escenario real, se procede a agrupar la información en sesiones, al igual que en el caso del teclado. Los resultados se pueden observar en la Tabla 5.7 y la Tabla 5.8. En la Figura 5.2 se muestra un ejemplo de los valores del *buffer* para un usuario concreto. Centrando la atención en la sesión 2, se puede observar que el umbral óptimo definido en este caso es ligeramente restrictivo pues el modelo clasifica como impostor dicha sesión a pesar de los valores bajos del *buffer* en ese instante. Esto hace que el sistema sea más seguro a la hora de detectar impostores pero también repercute en que el sistema considere a un usuario genuino como impostor en más ocasiones.

Tabla 5.7. Resultados del modelo de clasificación OC-SVM utilizando el *buffer* temporal para las dinámicas de ratón divididas en sesiones utilizando en el conjunto de test: FAR, FRR y EER (desviación típica)

Tamaño de sesión	N-grama	FAR	FRR	EER
5	1	0,352	0,357	0,373 (0,071)
5	2	0,376	0,388	0,337 (0,131)
5	3	0,376	0,360	0,381 (0,076)
5	6	0,366	0,386	0,296 (0,175)
10	1	0,237	0,225	0,242 (0,131)
10	2	0,292	0,306	0,279 (0,120)
10	3	0,329	0,325	0,327 (0,119)
10	6	0,285	0,381	0,278 (0,181)
20	1	0,151	0,151	0,143 (0,122)
20	2	0,232	0,232	0,220 (0,135)
20	3	0,274	0,246	0,240 (0,205)
20	6	0,258	0,366	0,248 (0,188)

En resumen, los mejores resultados para la RP específica propuesta en el caso de uso (chat) se obtienen modelando las dinámicas de teclado. En particular, los mejores resultados para este escenario se obtienen utilizando la distancia de Manhattan con un número de vecinos igual a 3 para el modelo de KNN y para sesiones de longitud 20. Por otro lado, para el caso de las dinámicas de ratón, los mejores resultados se obtienen utilizando una OC-SVM con n-gramas de longitud 1 y para sesiones de longitud 20. En ambos casos, tal y como ha quedado demostrado anteriormente, cuanto mayor es la duración de la sesión más información se considera y, por lo tanto, se obtienen mejores resultados.

5.1.5. Integración en los estándares de federación de identidades

La integración del flujo de trabajo propuesto depende del caso de uso en cuestión. Considerando los casos de uso vistos en la Sección 3.2, se precisan realizar modificaciones que se detallan a continuación y que se pueden ver esquematizadas en la Figura 5.3. El entorno de trabajo desarrollado implementa

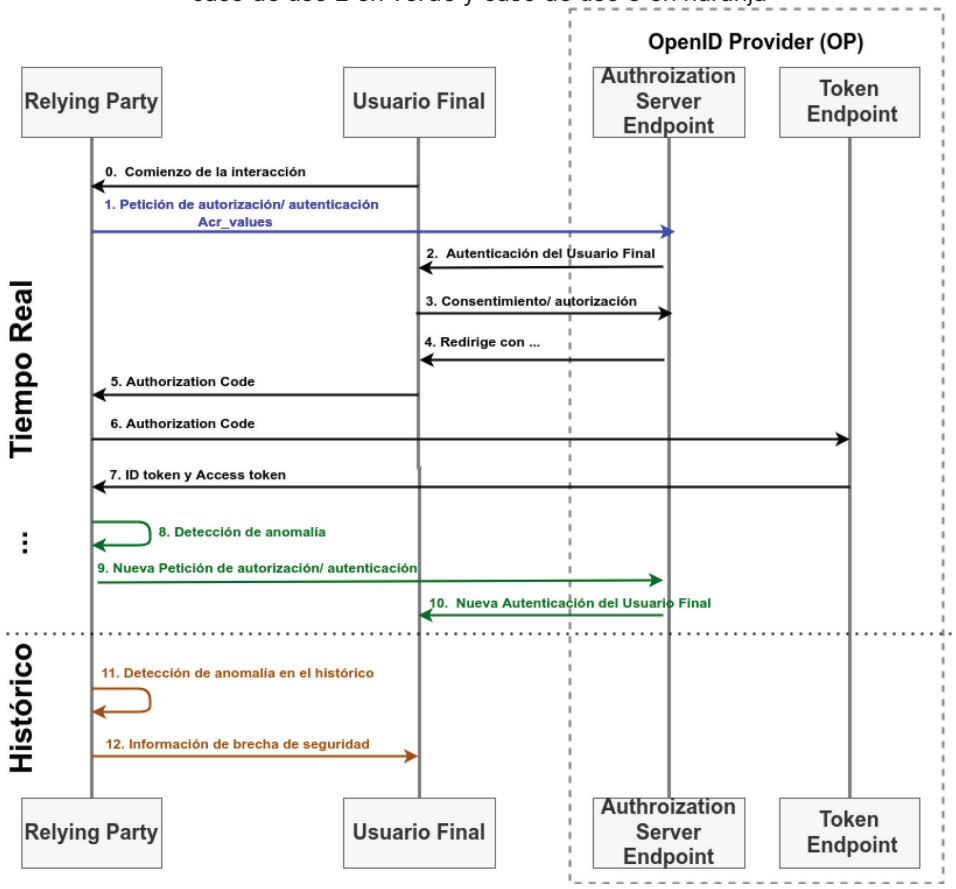
Tabla 5.8. Resultados del modelo de clasificación OC-SVM utilizando el *buffer* temporal para las dinámicas de ratón divididas en sesiones utilizando en el conjunto de validación: FAR y FRR

Tamaño de sesión	N-grama	FAR	FRR
5	1	0,353	0,355
5	2	0,381	0,410
5	3	0,430	0,426
5	6	0,425	0,451
10	1	0,307	0,329
10	2	0,341	0,372
10	3	0,370	0,380
10	6	0,386	0,462
20	1	0,256	0,277
20	2	0,295	0,356
20	3	0,355	0,385
20	6	0,363	0,368

el estándar federado OIDC (ver Sección 5.1.1), sin embargo, todos estos casos de uso son igualmente aplicables a cualquier estándar federado.

En el caso de uso 1 solo se requiere realizar una pequeña modificación en el flujo de autenticación. Esta modificación consiste en marcar como obligatorio el parámetro, actualmente considerado como opcional, *acr_values* de la *petición de autorización/autenticación*. De esta forma la RP debe utilizar este valor para especificar el LoA que requiere por parte del usuario final para completar el proceso de autenticación frente al IdP. Cuanto mayor sea el riesgo asociado a la petición de acceso, porque el modelo ha determinado una probabilidad alta de que sea un comportamiento anormal (impostor), mayor será el LoA requerido. De este mismo modo, el parámetro *acr_values* del *ID token* también se vuelve obligatorio. Así, el IdP debe informar a la RP del método utilizado para autenticar al usuario final cumpliendo con el LoA previamente determinado. Por la parte del IdP se han de contemplar diferentes métodos y procesos para autenticar a los usuarios. Todas estas consideraciones de implementación no afectan a la especificación federada, sino que han de solventarse a nivel de implementación de código por parte del IdP y la RP.

Figura 5.3. Modificaciones necesarias en el flujo de OpenId Connect para incorporar los casos de uso del flujo de trabajo propuesto. Caso de uso 1 en azul, caso de uso 2 en verde y caso de uso 3 en naranja



En el caso de uso 2 la RPs debe en primer lugar invalidar los *tokens* asociados a la sesión activa. Para ello, debe enviar dichos *tokens* al *Token Revocation Endpoint* [151]. De esta forma, la sesión activa del usuario queda invalidada. Posteriormente, la RP debe iniciar nuevamente el proceso enviado una nueva petición de autorización/autenticación al *Authorization Server Endpoint*, el cual volverá nuevamente a solicitar al usuario los autenticadores pertinentes. Este caso de uso se puede combinar con el caso de uso 1, pues la RP al detectar la anomalía puede solicitar un LoA mayor aumentando así los niveles de seguridad. Para este caso de uso, no se requiere ninguna modificación de los flujos de las especificaciones federadas. Se requiere una implementación a nivel de código por parte de la RP para forzar que se invaliden los *tokens* activos y se realice una nueva petición de ellos cada vez que los modelos

de autenticación continua determinen que se ha producido una anomalía de comportamiento y, por consiguiente, pueda estar sucediendo un ataque (p. ej. un secuestro de sesión). Esto desencadena una nueva petición de autorización/autenticación inicializando nuevamente el proceso.

En el caso de uso 3 no se ven afectados los estándares tradicionales de gestión de identidades federados. Nuevamente, implica implementar nuevos procedimientos en la RP para poder gestionar las situaciones adversas y procedimientos para comunicárselo al resto de agentes implicados en caso de que suceda un ataque de suplantación de identidad.

En resumen, para poder integrar el flujo de trabajo propuesto en los principales esquemas de gestión de identidades, el único cambio que habría que realizar sobre ellos es marcar como obligatorios dos parámetros (uno en el *ID token* y otro en el *access token*) que actualmente se encuentran como opcionales. De esta forma, se puede corroborar que la integración del flujo de trabajo es muy simple y, por consiguiente, facilita de forma notoria la integración e implantación del mismo por parte de multitud de RPs.

5.1.6. Eficiencia y análisis de seguridad

En las secciones anteriores se ha mencionado que las técnicas de UEBA pueden ser utilizadas en múltiples casos de uso y en diversas RPs de diferente naturaleza y dominios, ejecutadas sobre plataformas heterogeneas. Es por esto que uno de los puntos clave a analizar para lograr una adopción del flujo de trabajo propuesto es la eficiencia del mismo. De este modo, el consumo de recursos computacionales ha de ser mínimo y las posibles latencias introducidas no deben afectar al funcionamiento normal de una RP (en el caso de uso 2), o a la experiencia de usuario (en el caso de uso 1).

En este sentido, el tiempo medio de ejecución a la hora de comprar dos vectores de comportamiento utilizando el KNN y la distancia de Manhattan es de 0,000416 ± 0,000883 nanosegundos ejecutando sobre la máquina descrita en la Sección 5.1.1. En el caso de la OC-SVM, el tiempo medio de ejecución al comparar dos vectores de comportamiento es de 0,000040±0,0000923.

Tabla 5.9. Resultados para el ataque de robo de credenciales en el caso de uso 2

Tipo	FAR	FRR
Atributos estáticos	0	0
Dinámicas de teclado	0,104	0,082
Dinámicas de ratón	0,146	0,127

Estos tiempos corroboran que las técnicas de UEBA propuestas no afectan para la implantación e integración del flujo de trabajo para ninguno de los casos de uso propuestos, ya sean para autenticación estática o autenticación continua.

El último grupo de experimentos realizados se centran en analizar la mejora en cuanto a seguridad que ofrecen las técnicas de UEBA en el caso de uso más complejo. Recordemos que el caso de uso 2 requiere autenticación continua. Para lograr esto, se han incluido dos tipos de ataques. En primer lugar una suplantación de identidad por medio de un robo de credenciales. Los participantes en el experimento tuvieron que hacer públicas sus credenciales, de tal manera que todos los usuarios podían autenticarse como cualquier otro usuario, suplantando así su identidad. En segundo lugar, un secuestro de sesión. Los participantes tenían acceso a los diferentes dispositivos del resto de usuarios y, por lo tanto, tenían vía libre para poder hacerse pasar por cualquiera de ellos, nuevamente suplantando su identidad. Para evaluar este experimento, se han utilizado los mejores modelos obtenidos en la Sección 5.1.4.

Los resultados para el primer ataque se pueden observar en la Tabla 5.9. Se ha necesitado una media de 12,200 ± 4,176 interacciones para detectar a un impostor utilizando las dinámicas de teclado. Por otro lado, una media de 18,800 ± 2,081 interacciones se han necesitado para detectar a un impostor utilizando las dinámicas de ratón. Tal y como se ha mencionado en la Sección 5.1.4, utilizar atributos estáticos implica una clasificación perfecta debido al bajo número de participantes.

Los resultados para el segundo ataque, es decir, para el secuestro de sesión utilizando el dispositivo de la víctima, se pueden observar en la Tabla 5.10. Para las dinámicas de teclado se han necesitado una media de 13,100 ± 4,223 interacciones para detectar a un impostor, mientras que para las dinámicas de ratón 18,100 ± 2,714. En este caso, al utilizar el dispositivo de la víctima, los atributos estáticos no son útiles para detectar a ningún impostor pues siempre van a tener el mismo valor esperado.

Tabla 5.10. Resultados para el ataque de secuestro de sesión en el caso de uso 2

Tipo	FAR	FRR
Atributos estáticos	1	1
Dinámicas de teclado	0,149	0,104
Dinámicas de ratón	0,175	0,168

Como se puede observar, los resultados obtenidos para el primer ataque son mejores que para el segundo, por lo que, se puede afirmar que los modelos propuestos son mejores a la hora de detectar ataques de robo de identidad que ataques de secuestro de sesión.

5.2. Evaluación del método de combinación de la información

Hasta el momento ha quedado demostrado que es posible implementar el flujo de trabajo propuesto y, por consiguiente, integrar modelos de análisis de comportamiento dentro de los principales estándares de gestión de identidades federados. Sin embargo, los modelos de análisis de comportamiento desarrollados hasta el momento, aun funcionales, poseen multitud de limitaciones, entre las que cabe destacar la eficacia de los modelos. De aquí en adelante se evalúa el modelo de combinación de información propuesto que supera estas limitaciones.

5.2.1. Evaluación del método de combinación en el conjunto de datos UEBA

En esta sección se va a analizar la evaluación del modelo de combinación de información propuesto en el conjunto de datos UEBA. En primer lugar, se realiza el proceso de extracción de características. Para ello, en el caso del teclado se utilizan las características H1, H2, HP, PP, HH y PH descritas en la Sección 5.1.3. En el caso de las dinámicas de ratón, cada digrafo se ha agrupado en ventanas temporales de 5 segundos. Se han considerado el número de interacciones contenidas en la ventana temporal, el tiempo total transcurrido, la distancia, velocidad, ángulo y velocidad angular. De este modo se han calculado las medidas de dispersión (mínimo, máximo, media y desviación típica) para cada característica considerada anteriormente (tiempo transcurrido, distancia, etc). De esta forma, el vector de comportamiento obtenido, para el caso del ratón, contiene un total de 21 características (número de interacciones contenidas en la ventana temporal y cuatro medidas de dispersión de cinco variables). En resumen, para cada usuario se obtienen dos vectores (uno para cada fuente de información) que contiene la información de comportamiento.

Con el objetivo de entrenar y posteriormente poder evaluar el método, la información, ya procesada para cada usuario, se separa en tres conjuntos denominados: entrenamiento, test y validación. El conjunto de entrenamiento representa el 70 % de la información total del usuario objetivo, es decir, contiene únicamente muestras genuinas. Por otro lado, el conjunto de test está formado por muestras tanto genuinas como pertenecientes a usuarios impostores. Para el caso de las muestras genuinas, del 30 % restante (no

seleccionado para entrenamiento) se selecciona un total del 60 % (es decir, un 18 % del total de muestras genuinas). Para el caso de las muestras de impostores, el mismo número de muestras que para el caso de las genuinas son aleatoriamente seleccionadas del resto de usuarios. Además, se asegura que esta información de los usuarios impostores contenga muestras de ambas fuentes de información (teclado y ratón). Esto significa que el resto de los usuarios actúan como impostores a la hora de evaluar a un usuario específico. Finalmente, el conjunto de validación está formado por el 12 % restante de muestras genuinas y por el mismo número de muestras de usuarios impostores siguiendo los mismos criterios que para el conjunto de test.

En primer lugar, para cada usuario, el conjunto de entrenamiento se utiliza para estandarizar el resto de los conjuntos de datos. Esto se logra calculando la media y desviación típica para cada una de las fuentes de información y aplicando la fórmula que sigue:

$$z_i = \frac{x_i - \overline{X}}{S_x}$$

donde z_i es el valor estandarizado para la muestra i, x_i es el valor original de la muestra i, $\overline{X}$ es la media del conjunto de muestras *y* S_x es la desviación típica.

Una vez estandarizados los datos, se utiliza el conjunto de entrenamiento para entrenar el algoritmo de RTE. El resultado de este proceso es la secuencia de símbolos que representa el comportamiento de un usuario. Esta secuencia, se separa en n-gramas utilizando el parámetro *NGL*, obteniendo múltiples secuencias que representan la información de comportamiento de un usuario concreto. Estos n-gramas se utilizan para generar la matriz de distancias, comparándolas en pares utilizando el algoritmo de alineamiento de secuencias de ADN. A continuación, la matriz de distancias se utiliza para entrenar el algoritmo de DBSCAN. El resultado son los núcleos de comportamiento que representan el conocimiento adquirido para un usuario concreto y, por tanto, concluyendo el proceso de entrenamiento.

Posteriormente, las muestras de test (ya estandarizadas) se evalúan sobre el RTE previamente entrenado. De esta forma y al igual que en el caso anterior, se obtiene la secuencia de símbolos y se divide en n-gramas. Estos n-gramas se comparan uno a uno contra todos los n-gramas contenidos en los núcleos de comportamiento utilizando el algoritmo de alineamiento de secuencias de ADN. El resultado son múltiples vectores de distancias. Cada vector contiene la distancia de cada muestra a los núcleos de comportamiento. Estos vectores se utilizan para entrenar el modelo de riesgos, agrupándose en ventanas temporales en función del parámetro *WinSize* y pudiendo fijar el umbral óptimo de decisión.

Por último, el conjunto de validación se utiliza para evaluar todo el proceso con muestras que el método propuesto no ha considerado nunca. Este conjunto de datos permite analizar la capacidad de generalización y la efectividad del método propuesto. Esto se debe a que, al evaluar este conjunto de datos, todos los modelos que se integran en el método, así como el umbral de decisión, han sido previamente entrenados y fijados.

Una vez obtenidos los tres conjuntos de datos, se evalúa el método teniendo en cuenta las tres casuísticas asociadas a las fuentes de información. Esto es, considerando únicamente el teclado, considerando únicamente el ratón y considerando la combinación de ambas fuentes de información. Los resultados se pueden observar en las Tablas 5.11, 5.12 y 5.13 respectivamente. Los resultados mostrados para cada métrica de evaluación se corresponden con la media obtenida para todos los usuarios.

Para todas las casuísticas de fuentes de información, los valores de EER, FAR y FRR disminuyen (mejoran los resultados) cuando los valores de los parámetros *NGL* o *WinSize* aumentan. Esto se debe a que, cuanto mayor son los parámetros, más información se está considerando para realizar una predicción. De este modo, los valores máximos de EER para cada caso de uso (0,457, 0,438 y 0,397 respectivamente) se obtienen para los valores *NGL* = 5 y *WinSize* = 5. Por otro lado, se obtienen predicciones casi perfectas cuando se fijan los parámetros a los valores *NGL* = 30 y *WinSize* = 100. Unicamente fijando el parámetro *WinsS ize* a 100, se obtienen buenos resultados independientemente del valor de *NGL*. Sin embargo, estos resultados se vuelven cada vez más robustos a medida que aumenta el parámetro *NGL* para el conjunto de validación.

Tabla 5.11. Resultados obtenidos para las dinámicas de teclado utilizando el método de combinación en el conjunto de datos UEBA. El sufijo _val se refiere a conjunto de validación. Los mejores resultados obtenidos para cada métrica se muestran en negrita

NGL	WinSize	EER	FAR	FRR	FAR_val	FRR_val
5	5	0.457	0.457	0.457	0.442	0.397
5	10	0.423	0.423	0.423	0.439	0.392
5	20	0.356	0.356	0.366	0.420	0.287
5	50	0.268	0.239	0.282	0.323	0.207
5	100	**0.000**	**0.000**	0.048	0.250	0.062
10	5	0.384	0.381	0.386	0.303	0.453

(*continuar*)

Tabla 5.11. Resultados obtenidos para las dinámicas de teclado utilizando el método de combinación en el conjunto de datos UEBA. El sufijo _val se refiere a conjunto de validación. Los mejores resultados obtenidos para cada métrica se muestran en negrita (*continuar*)

NGL	WinSize	EER	FAR	FRR	FAR_val	FRR_val
10	10	0.343	0.333	0.343	0.189	0.414
10	20	0.276	0.276	0.286	0.122	0.250
10	50	0.059	0.059	0.074	0.015	0.167
10	100	**0.000**	**0.000**	0.047	**0.000**	0.186
20	5	0.304	0.301	0.305	0.297	0.141
20	10	0.226	0.228	0.226	0.230	0.030
20	20	0.145	0.145	0.147	0.163	0.019
20	50	0.050	0.051	0.050	0.043	0.027
20	100	**0.000**	**0.000**	0.011	0.003	0.026
30	5	0.267	0.267	0.270	0.236	0.254
30	10	0.177	0.175	0.178	0.142	0.097
30	20	0.091	0.090	0.091	0.038	0.038
30	50	0.006	**0.000**	**0.008**	**0.000**	**0.002**
30	100	**0.000**	**0.000**	0.012	**0.000**	0.009

Los mejores resultados para test y validación simultáneamente se obtienen para el caso de uso de combinación de la información. Esto corrobora que combinar información de múltiples fuentes ayuda a mejorar la eficacia de los sistemas de autenticación continua que solo consideran una única fuente.

Tabla 5.12. Resultados obtenidos para las dinámicas de ratón utilizando el método de combinación en el conjunto de datos UEBA. El sufijo _val se refiere a conjunto de validación. Los mejores resultados obtenidos para cada métrica se muestran en negrita

NGL	WinSize	EER	FAR	FRR	FAR_val	FRR_val
5	5	0.438	0.438	0.438	0.308	0.390
5	10	0.392	0.392	0.396	0.234	0.359
5	20	0.347	0.343	0.351	0.146	0.280
5	50	0.265	0.265	0.265	0.031	0.076

Tabla 5.12. Resultados obtenidos para las dinámicas de ratón utilizando el método de combinación en el conjunto de datos UEBA. El sufijo _val se refiere a conjunto de validación. Los mejores resultados obtenidos para cada métrica se muestran en negrita (*continuar*)

NGL	WinSize	EER	FAR	FRR	FAR_val	FRR_val
5	100	0.034	0.024	0.065	0.506	0.513
10	5	0.354	0.354	0.358	0.329	0.306
10	10	0.306	0.302	0.306	0.261	0.176
10	20	0.202	0.198	0.202	0.129	0.129
10	50	0.090	0.090	0.094	8.000	**0.000**
10	100	**0.000**	**0.000**	0.031	0.500	0.500
20	5	0.147	0.139	0.151	0.133	0.223
20	10	0.077	0.073	0.081	0.112	0.050
20	20	0.034	0.030	0.034	0.053	**0.000**
20	50	0.000	**0.000**	0.016	**0.000**	**0.000**
20	100	**0.000**	**0.000**	0.026	**0.000**	**0.000**
30	5	0.090	0.090	0.090	0.080	0.049
30	10	0.025	0.025	0.025	0.013	0.006
30	20	0.004	0.000	0.012	**0.000**	0.011
30	50	**0.000**	**0.000**	**0.007**	**0.000**	0.041
30	100	**0.000**	**0.000**	0.011	**0.000**	0.092

5.2.2. Evaluación del método de combinación en el conjunto de datos TWOS

En esta sección se evalúa el método de combinación de información propuesto sobre el conjunto de datos TWOS. El conjunto de datos TWOS [105] se recogió durante la competición organizada por la Universidad de Singapur de tecnología y diseño en marzo de 2017. Los datos provienen de seis fuentes de información: teclado, ratón, tráfico de red, registros *Simple Mail*

Transfer Protocol (SMTP), información de inicio de sesión e información de la máquina anfitriona. Además, posee información relacionada a un test psicológico de personalidad realizado a cada uno de los participantes. En total,

Tabla 5.13. Resultados obtenidos para la combinación de información en el conjunto de datos UEBA. El sufijo _val se refiere a conjunto de validación. Los mejores resultados obtenidos para cada métrica se muestran en negrita

NGL	WinSize	EER	FAR	FRR	FAR_val	FRR_val
5	5	0.397	0.397	0.397	0.342	0.486
5	10	0.364	0.364	0.366	0.294	0.501
5	20	0.320	0.317	0.320	0.273	0.484
5	50	0.229	0.223	0.232	0.131	0.489
5	100	0.112	0.115	0.112	0.049	0.448
10	5	0.289	0.284	0.292	0.322	0.329
10	10	0.219	0.222	0.219	0.284	0.296
10	20	0.141	0.138	0.144	0.154	0.263
10	50	0.029	0.024	0.032	0.019	0.100
10	100	**0.000**	**0.000**	0.007	**0.000**	0.019
20	5	0.167	0.167	0.169	0.117	0.146
20	10	0.080	0.075	0.085	0.052	0.070
20	20	0.006	0.006	0.010	0.016	0.015
20	50	**0.000**	**0.000**	0.006	**0.000**	0.008
20	100	**0.000**	**0.000**	0.008	**0.000**	0.011
30	5	0.128	0.128	0.131	0.217	0.116
30	10	0.054	0.068	0.054	0.165	0.030
30	20	0.020	0.014	0.023	0.098	0.001
30	50	**0.000**	**0.000**	0.006	**0.000**	0.008
30	100	**0.000**	**0.000**	**0.004**	**0.000**	**0.000**

veinticuatro usuarios participaron en la recogida de datos durante un periodo de cinco días.

Al igual que para el conjunto de datos UEBA, se ha seleccionado la información que proviene de las fuentes de información del teclado y del ratón. Esta información se ha procesado y dividido en los conjuntos de entrenamiento, test y validación siguiendo las mismas directrices marcadas en el caso anterior. En definitiva, se han extraído los núcleos de comportamiento y se ha

Tabla 5.14. Resultados obtenidos para las dinámicas de teclado utilizando el método de combinación en TWOS. El sufijo _val se refiere a conjunto de validación. Los mejores resultados obtenidos para cada métrica se muestran en negrita

NGL	WinSize	EER	FAR	FRR	FAR_val	FRR_val
5	5	0.414	0.413	0.415	0.407	0.414
5	10	0.386	0.386	0.386	0.375	0.384
5	20	0.341	0.343	0.341	0.346	0.353
5	50	0.269	0.269	0.269	0.268	0.233
5	100	0.214	0.219	0.214	0.189	0.131
10	5	0.362	0.362	0.362	0.380	0.382
10	10	0.337	0.335	0.339	0.340	0.320
10	20	0.305	0.304	0.307	0.282	0.237
10	50	0.217	0.217	0.217	0.180	0.126
10	100	0.133	0.133	0.134	0.044	0.052
20	5	0.317	0.318	0.317	0.346	0.302
20	10	0.259	0.257	0.259	0.305	0.245
20	20	0.189	0.189	0.191	0.227	0.172
20	50	0.080	0.078	0.082	0.110	0.072
20	100	0.008	0.008	0.008	0.018	0.019
30	5	0.299	0.299	0.299	0.293	0.288
30	10	0.231	0.231	0.232	0.222	0.224
30	20	0.156	0.156	0.156	0.139	0.139
30	50	0.055	0.054	0.055	0.060	0.052
30	100	**0.007**	**0.007**	**0.007**	**0.000**	**0.015**

entrenado el modelo de riesgos con el objetivo de poder evaluar los tres casos de uso.

Los resultados se pueden observar en las Tablas 5.14, 5.15 y 5.16, respectivamente. Al igual que en la sección anterior, se han considerado todas las posibles combinaciones de los parámetros *NGL* y *WinSize*. Los resultados obtenidos para cada métrica se corresponden con la media para todos los usuarios.

Los resultados tanto para cada fuente de información por separado, como para la combinación de ambas fuentes son óptimos. Al igual que en la sección anterior, los resultados del método propuesto mejoran cuando se considera más información, es decir, cuando los valores de los parámetros *NGL* y *WinSize* aumentan. En el caso de uso del teclado, los valores de EER van desde 0,414 hasta 0,007, obteniendo valores de FAR y FRR en

Tabla 5.15. Resultados obtenidos para las dinámicas de ratón utilizando el método de combinación en TWOS. El sufijo _val se refiere a conjunto de validación. Los mejores resultados obtenidos para cada métrica se muestran en negrita

NGL	WinSize	EER	FAR	FRR	FAR_val	FRR_val
5	5	0.433	0.433	0.434	0.400	0.432
5	10	0.404	0.403	0.404	0.375	0.400
5	20	0.371	0.370	0.372	0.313	0.351
5	50	0.317	0.316	0.318	0.252	0.286
5	100	0.266	0.265	0.266	0.205	0.260
10	5	0.420	0.419	0.420	0.406	0.394
10	10	0.392	0.392	0.393	0.369	0.356
10	20	0.355	0.354	0.355	0.290	0.315
10	50	0.285	0.285	0.285	0.263	0.209
10	100	0.180	0.176	0.180	0.250	0.126
20	5	0.377	0.376	0.379	0.362	0.388
20	10	0.336	0.335	0.337	0.311	0.347
20	20	0.286	0.285	0.288	0.255	0.287
20	50	0.174	0.173	0.174	0.153	0.211
20	100	0.088	0.086	0.101	0.086	0.117
30	5	0.359	0.357	0.361	0.338	0.352
30	10	0.313	0.312	0.314	0.308	0.302
30	20	0.250	0.250	0.251	0.237	0.243
30	50	0.136	0.136	0.137	0.141	0.146
30	100	**0.051**	**0.049**	**0.089**	**0.047**	**0.073**

validación de 0,407 y 0,414 respectivamente para la peor combinación de parámetros y, 0,000 y 0,015 para la mejor combinación de los mismos. Por otro lado, en el caso de uso del ratón, los resultados son ligeramente peores. Así, los valores de EER van desde 0,433 hasta 0,051, obteniendo valores de FAR y FRR para validación de 0,400 y 0,432 respectivamente para la peor combinación de parámetros y, 0,047 y 0,073 para la mejor

Tabla 5.16. Resultados obtenidos para la combinación de información en TWOS. El sufijo _val se refiere a conjunto de validación. Los mejores resultados obtenidos para cada métrica se muestran en negrita

NGL	WinSize	EER	FAR	FRR	FAR_val	FRR_val
5	5	0.407	0.407	0.407	0.409	0.410
5	10	0.380	0.380	0.380	0.381	0.375
5	20	0.335	0.335	0.335	0.343	0.321
5	50	0.257	0.257	0.257	0.270	0.228
5	100	0.177	0.177	0.176	0.197	0.150
10	5	0.359	0.359	0.359	0.371	0.349
10	10	0.320	0.319	0.319	0.335	0.307
10	20	0.263	0.262	0.263	0.289	0.245
10	50	0.169	0.168	0.169	0.213	0.159
10	100	0.082	0.082	0.082	0.140	0.099
20	5	0.295	0.295	0.296	0.310	0.306
20	10	0.226	0.226	0.226	0.267	0.230
20	20	0.156	0.156	0.157	0.206	0.157
20	50	0.066	0.065	0.067	0.098	0.080
20	100	0.022	0.022	0.022	0.041	0.018
30	5	0.268	0.268	0.268	0.282	0.285
30	10	0.201	0.200	0.201	0.227	0.211
30	20	0.121	0.122	0.121	0.168	0.110
30	50	0.038	0.037	0.038	0.076	0.020
30	100	**0.006**	**0.006**	**0.006**	**0.014**	**0.004**

combinación de los mismos. Los mejores resultados de forma global se obtienen para la combinación de ambas fuentes de información. En este caso, los valores de EER se encuentran en el rango 0,407 y 0,006 llegando a valores de FAR y FRR en el conjunto de validación de 0,409 y 0,410 respectivamente para la peor combinación de parámetros y de 0,0014 y 0,004 para la mejor combinación.

En el caso particular del teclado, cada símbolo de la secuencia (digrafo) tiene un tiempo medio de ejecución de 0,227 segundos. En el caso de las dinámicas de movimiento del ratón, cada símbolo de la secuencia representa una ventana temporal de 5 segundos. Esto significa que, considerando un *NGL* = 10, los n-gramas contienen aproximadamente 2,27 segundos de información para las dinámicas de teclado y 50 segundos de información para las dinámicas de ratón. Del mismo modo, seleccionando un *WinSize* = 100 para realizar una predicción se considera 22,7 segundos de información histórica para el teclado y 500 segundos para el ratón. En el caso de la combinación de información, una secuencia promedio para todos los usuarios contiene un 73 % de símbolos pertenecientes al teclado, mientras que un 27 % pertenecen al ratón. Considerando un *NGL* = 10, el vector promedio contendrá por lo tanto 7 símbolos del teclado y 3 del ratón. Esto se traduce en que un vector promedio contiene de media 16,589 segundos de información. Tomando un *WinSize* = 100, se considerará de media un total de 151,571 segundos de información histórica para realizar una predicción. A pesar de que se consideré toda esta información histórica, las predicciones se realizan para cada interacción del usuario, es decir, para cada nuevo símbolo independientemente de si proviene del teclado o del ratón. De este modo, las predicciones se realizan de media cada 0,227 segundos para el teclado y 5 segundos para el ratón. Esto se debe a que, cuando se evalúan los n-gramas, la secuencia adyacente es exactamente igual a la secuencia anterior a excepción del primer y último símbolo (ver Figura 4.9).

Finalmente, los algoritmos de SVM y RF se han seleccionado como algoritmos representativos del estado del arte frente a los que comparar el método propuesto. De este modo, para entrenar estos algoritmos se ha realizado un preprocesamiento de los datos para que consideren los parámetros *NGL* y *Winsize*, tal como lo hace el método propuesto. En primer lugar, los datos en bruto se han agrupado en subgrupos acorde al parámetro *NGL*. Posteriormente, cada uno de estos modelos ha sido entrenado utilizando una búsqueda en cuadrículas para fijar los hiperparámetros. Cada modelo devuelve la probabilidad de pertenecer a la clase genuina y a la clase impostora. Para obtener el riesgo, se selecciona la probabilidad devuelta de pertenecer a la

clase impostora. Estas probabilidades se agrupan en ventanas temporales utilizando el parámetro *WinSize*. Posteriormente, se aplica MME para suavizar la curva de riesgo siguiendo las directrices marcadas por el método propuesto.

En la Tabla 5.17 se comparan los resultados obtenidos para el método propuesto frente a los algoritmos de SVM y RF, así como con otras propuestas del estado del arte. Los resultados mostrados para [96] son los obtenidos para el algoritmo de 2D-CNN. Los resultados mostrados para [104] son los obtenidos para el algoritmo de SVM.

Los resultados mostrados para el método propuesto son los obtenidos para el conjunto de parámetros tal que, los resultados en validación son comparables a los resultados del resto de propuestas o algoritmos. Esto es, los valores de *NGL* y *WinSize* son (30,100), (20,100), (5,100) y (20,50), respectivamente. Estos mismos parámetros han sido utilizados para entrenar los algoritmos de SVM y RF. El primer bloque de resultados se corresponde con el uso de dinámicas de teclado, el segundo bloque se corresponde con el uso de dinámicas de ratón y el tercer bloque se corresponde con la combinación de la información de teclado y de ratón. Finalmente, en el cuarto bloque se compara la combinación de información de la propuesta [104] en la que se utiliza información de contexto. Puesto que esta información no está disponible públicamente y pertenece a un conjunto de datos externo y privado, en este bloque se demuestra que utilizando el método propuesto se pueden obtener resultados comprables e incluso mejores en algunos ámbitos, única y exclusivamente utilizando información del teclado y del ratón.

Comparando los resultados de [96] con los obtenidos en la Tabla 5.15, los valores de *NGL* = 30 y *WinSize* = 50 muestran resultados similares para el método propuesto. Sin embargo, si se utiliza más información (*NGL* = 20 y *WinSize* = 100 o *NGL* = 30 y *WinSize* = 100) los resultados obtenidos por el método mejoran considerablemente. Lo mismo sucede en la combinación de información para la propuesta de [104], donde los resultados pueden ser igualados fijando los valores de *NGL* y *WinSize* a (5,100), (10,50), (20,20) o (30,20) respectivamente (ver Tabla 5.16). Cuando en esa propuesta se considera información externa de contexto, los resultados pueden ser igualados o incluso mejorados utilizando únicamente información de teclado y de ratón por el método aquí propuesto fijando los parámetros a (20,50), (20,100), (30,50) o (20,100). Por último, considerando el EER, los algoritmos SVM y RF obtienen resultados satisfactorios. Sin embargo, ambos métodos son superados por el método propuesto en igualdad de condiciones.

Tabla 5.17. Comparación de los resultados obtenidos con otras propuestas y algoritmos del estado del arte. *T* y *M* representan las dinámicas de teclado y de ratón respectivamente. *C* se refiere a información de contexto recogida de un conjunto de datos externos. RF+RF representa un modelo de combinación de información en el que se utiliza el algoritmo de RF de forma independiente para cada fuente de datos. FI se refiere a fuente de información. Exac se refiere a exactitud. Espec representa la especificación. El sufijo _val se refiere a conjunto de validación. Los mejores resultados obtenidos para métrica en cada bloque de experimentos se muestran en negrita

Trabajo	FI	EER	FAR_val	FRR_val	F1-_val	Exac_val	VPN_val	Espec_val
SVM	T	0.136	0.158	0.151	0.850	0.845	0.858	0.842
RF	T	0.084	0.093	0.174	0.860	0.865	0.819	0.907
Our	T	**0.007**	**0.000**	**0.015**	**0.968**	**0.979**	**0.945**	**0.993**
SVM	R	0.171	0.180	**0.063**	0.882	0.877	**0.955**	0.820
RF	R	0.109	0.141	0.081	0.890	0.888	0.925	0.859
[96]	R	0.130	0.136	0.149	-	-	-	-
Our	R	**0.088**	**0.086**	0.117	**0.900**	**0.909**	0.888	**0.914**
RF+RF	T + R	0.180	0.228	0.169	0.814	0.800	0.861	0.772
[104]	T + R	-	-	-	0.806	0.751	**0.932**	0.710
Our	T + R	**0.177**	**0.197**	**0.150**	**0.828**	**0.826**	0.856	**0.803**
RF+RF	T + R	0.121	0.126	0.152	0.860	0.860	0.848	0.874
[104]	T + R + C	-	-	-	**0.914**	**0.915**	0.874	**0.912**
Our	T + R	**0.066**	**0.098**	**0.080**	0.912	**0.915**	**0.921**	0.902