MINISTÉRIO DA DEFESA
EXÉRCITO BRASILEIRO
DEPARTAMENTO DE CIÊNCIA E TECNOLOGIA
INSTITUTO MILITAR DE ENGENHARIA
CURSO DE GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

VICTOR VILLAS BÔAS CHAVES
LUCAS SOUSA MEIRELES
CLAUDIO CAVALCANTE BONFIM JÚNIOR

AGENTES INTELIGENTES NO RECONHECIMENTO DE USUÁRIOS DE SISTEMAS COMPUTACIONAIS BASEADO NA DINÂMICA DE DIGITAÇÃO

Rio de Janeiro
2017

INSTITUTO MILITAR DE ENGENHARIA

VICTOR VILLAS BÔAS CHAVES
LUCAS SOUSA MEIRELES
CLAUDIO CAVALCANTE BONFIM JÚNIOR

AGENTES INTELIGENTES NO RECONHECIMENTO DE USUÁRIOS DE SISTEMAS COMPUTACIONAIS BASEADO NA DINÂMICA DE DIGITAÇÃO

Projeto de Fim de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Engenheiro de Computação.

Orientador: Prof. Ronaldo Ribeiro Goldschmidt - D.Sc.

Rio de Janeiro
2017

INSTITUTO MILITAR DE ENGENHARIA
Praça General Tibúrcio, 80 - Praia Vermelha
Rio de Janeiro - RJ CEP 22290-270

006.3 Chaves, Victor Villas Bôas
C512a Agentes Inteligentes no Reconhecimento de Usuários de Sistemas Computacionais Baseado na Dinâmica de Digitação / Victor Villas Bôas Chaves, Lucas Sousa Meireles, Claudio Cavalcante Bonfim Júnior, orientado por Ronaldo Ribeiro Goldschmidt - Rio de Janeiro: Instituto Militar de Engenharia, 2017.

37p.: il.

Projeto de Fim de Curso (graduação) - Instituto Militar de Engenharia, Rio de Janeiro, 2017.

1. Curso de Graduação em Engenharia de Computação - projeto de fim de curso. 1. Autenticação. I. Goldschmidt, Ronaldo Ribeiro. II. Título. III. Instituto Militar de Engenharia.

INSTITUTO MILITAR DE ENGENHARIA

VICTOR VILLAS BÔAS CHAVES
LUCAS SOUSA MEIRELES
CLAUDIO CAVALCANTE BONFIM JÚNIOR

AGENTES INTELIGENTES NO RECONHECIMENTO DE USUÁRIOS DE SISTEMAS COMPUTACIONAIS BASEADO NA DINÂMICA DE DIGITAÇÃO

Projeto de Fim de Curso apresentado ao Curso de Graduação em Engenharia de Computação do Instituto Militar de Engenharia, como requisito parcial para a obtenção do título de Engenheiro de Computação.

Orientador: Prof. Ronaldo Ribeiro Goldschmidt - D.Sc.

Aprovado em 28 de Setembro de 2017 pela seguinte Banca Examinadora:

Prof. Ronaldo Ribeiro Goldschmidt - D.Sc. do IME - Presidente

Maj Humberto Henrique de Arruda - M.Sc. do IME

Prof. Ricardo Choren Noya - D.Sc. do IME

Rio de Janeiro
2017

> "What to do if you find yourself stuck with no hope of rescue: Consider yourself lucky that life has been good to you so far. Alternatively, if life hasn't been good to you so far, which given your present circumstances seems more likely, consider yourself lucky that it won't be troubling you much longer. "

DOUGLAS ADAMS, THE HITCHHIKER'S GUIDE TO THE GALAXY

SUMÁRIO

LISTA DE ILUSTRAÇÕES

RESUMO

Balancear autenticação de usuários com simplicidade de uso é um problema em aberto da segurança da informação. Uma abordagem promissora é baseada no uso de características biométricas, sendo dentre elas as de menor impacto de usabilidade as características comportamentais. A análise de dinâmica de digitação como característica comportamental é uma solução promissora para os diversos problemas da área de autenticação biométrica. Neste trabalho é proposto um serviço de autenticação de usuários via dinâmica de digitação, de forma a simplificar a adoção desta característica comportamental e manter o poder de configuração e a transparência. Isto é alcançado ao modelar um serviço externo que pode ser consultado pelos sistemas que desejem fazer a autenticação. O protótipo desenvolvido demonstra a facilidade de implantação desse serviço e o experimento realizado com dados públicos comprova a eficácia do modelo de aprendizado escolhido, demonstrando performance equiparável aos trabalhos relacionados de métodos semelhantes.

ABSTRACT

Balancing user authentication and usage simplicity is an open problem in information security. A promising approach is based on the use of biometric characteristics, with the least impact of usability being the behavioral characteristics. The analysis of keystroke dynamics as a behavioral characteristic is a promising solution to the various problems in biometric authentication. This work proposes a service of authentication of users via typing dynamics, in order to simplify the adoption of this behavioral characteristic and maintain configuration power and transparency. This is achieved by modeling an external service that can be queried by systems that wish to authenticate. The developed prototype demonstrates the ease of implementation of this service and the experiment performed with public data proves the effectiveness of the chosen learning model, demonstrating performance similar to the related work of similar methods.

1 INTRODUÇÃO

1.1 CONTEXTO E MOTIVAÇÃO

Autenticação é o processo de verificar positivamente a identidade de um usuário, dispositivo ou outra entidade em um sistema computacional, em geral como prerrequisito de permissão de acesso a recursos. A autenticação de uma entidade alcança verificação positiva ao corretamente corresponder a uma forma de identificação previamente definida (O'GORMAN, 2003).

Essa identificação pode ocorrer de diferentes formas, sendo as mais comuns um mecanismo de senha ou PIN (*Personal Identification Number*) baseado em algo que a entidade sabe, e um mecanismo de identificação via *token* baseado em algo que a entidade possui. (HU et al., 2008)

É conveniente discernir a diferença entre autenticação de dispositivos (máquinas) e usuários (pessoas). A autenticação de pessoas, em geral, envolve diversas dificuldades em função da usabilidade e interface para o usuário (O'GORMAN, 2003). *Tokens* podem ser roubados ou perdidos, enquanto *PINs* seguros são mais difíceis de lembrar (HU et al., 2008).

Por apresentar dificuldades de interface especiais, a atenção no estudo de sistemas de autenticação se concentra em usuários humanos. O uso de senhas que pessoas possam lembrar é por si só considerado um fator de risco, sendo portanto recomendável introduzir mecanismos secundários de autenticação que não causem queda de usabilidade significativa (MORRIS; THOMPSON, 1979).

Dentre as alternativas de métodos de autenticação, o uso de características biométricas se destaca por sua natureza individual e difícil falsificação. Utilizados principalmente em sistemas de autorização e autenticação em ambientes físicos, um sistema biométrico se baseia em um sensor consistente de assinatura biométrica (ED DAWSON et al., 2003).

As características biométricas mais frequentemente mencionadas são as fisiológicas, porém seu emprego traz diversos fatores complicantes relacionados à amostragem via sensores adicionais como a diminuição da usabilidade do sistema, custo de implementação ou manutenção e potencial ponto de falha do dispositivo sensor. Uma alternativa é o uso de características biométricas de comportamento, como padrões comportamentais expressos naturalmente pelo usuário (MOSKOVITCH et al., 2009).

As vantagens do uso de características comportamentais incluem a possibilidade de amostragem silenciosa, maior variabilidade do grau de confiança e a transparência do mecanismo para o usuário. Em particular, sistemas providos pela *Web* em geral possuem uma uniformidade de interface que permite a coleta de vários padrões comportamentais durante todo o uso do sistema sem a necessidade de instrumentação física adicional (MOSKOVITCH et al., 2009).

Além das vantagens do ponto de vista de usabilidade e interface, a análise de características biométricas combate um problema fundamental da autenticação tradicional baseada em *PINs* ou *tokens*: discernir se o usuário identificado não é um agente que tenha roubado as credenciais de acesso da pessoa verdadeira associada ao usuário no sistema. Características comportamentais em particular são vistas como as mais difíceis de serem perdidas, imitadas ou roubadas (HU et al., 2008).

Um caso particular de uso de características biométricas comportamentais é a análise de dinâmica de digitação, baseada em uma ampla amostragem das características temporais e sequenciais que descrevem as atividades das teclas quando são pressionadas ou liberadas conforme uma pessoa utiliza o teclado do computador (HU et al., 2008).

Embora características fisiológicas tenham tido um sucesso maior do que as biométricas por serem mais consistentes temporalmente (isto é, menor variância entre amostragens consecutivas), nos últimos anos diversos métodos de análise dos padrões de digitação tem resultado em soluções reais e patentes em sistemas de segurança, mostrando que a aplicabilidade destas características já é possível (BERGADANO et al., 2002).

Existem os diversos estudos voltados à investigação de reconhecimento de usuários baseado em dinâmica de digitação (como Bergadano et al. (2002), Hu et al. (2008), Lau et al. (2004) e Gunetti e Picardi (2005)). No entanto, são poucas as iniciativas para disponibilizar comercialmente tal funcionalidade.

1.2 OBJETIVOS

Esse trabalho tem como objetivo geral a especificação e construção de um sistema de autenticação baseado em dinâmica de digitação do usuário para ser disponibilizado como serviço externo. Nesta direção, o presente projeto conta como objetivos específicos:

- Analisar os tipos de informação que se pode extrair a partir dos padrões de digitação de um indivíduo;
- Modelar a combinação das informações extraídas utilizando algoritmos de aprendi-

zado de máquina;

- Sistematizar um mecanismo de coleta de amostras que permita o treinamento dos modelos escolhidos;
- Desenvolver uma biblioteca que facilite sua integração a outros componentes de serviços *Web*;
- Definir uma arquitetura de sistema para implantação dos mecanismos de coleta e autenticação definidos.

1.3 CONTRIBUIÇÕES ESPERADAS

Espera-se desse projeto a produção de um mecanismo de autenticação que age de forma completamente transparente ao usuário, como um serviço complementar de autenticação do mesmo, com a finalidade de impedir fraudes de acesso às aplicações do cliente. O projeto visa, assim, suprir a necessidade de métodos de autenticação de baixo impacto de usabilidade e fácil implantação.

1.4 MÉTODO

O desenvolvimento do serviço de autenticação foi feito de forma paralela ao da biblioteca. Por um lado a biblioteca é responsável por coletar os dados de dinâmica de digitação no navegador de um usuário, enquanto o serviço recebe esses dados e faz a verificação de identidade.

Após a arquitetura do sistema ter sido definida e implementada, foram experimentados modelos de aprendizado de máquina em um serviço protótipo para prova de conceito. Durante esse experimento pôde ser feita a coleta de dados e ao mesmo tempo avaliar a usabilidade da biblioteca.

1.5 ORGANIZAÇÃO DO TEXTO

No capítulo 2 são introduzidos os conceitos necessários para a modelagem conceitual de um sistema de autenticação por dinâmica de digitação, em particular discutindo os modelos de aprendizado de máquina utilizados (seção 2.1) e características comportamentais analisadas (seção 2.2). No capítulo 3 são discutidos trabalhos anteriores na área.

No capítulo 4 é apresentada uma arquitetura de sistema de autenticação isolado, para implantação do método apresentado. Na seção 4.1 é definido o modelo de autenticação,

especificando o fluxo de informações desde a coleta até a decisão de um grau de confiança de identidade, enquanto em 4.2 é demonstrada uma possível implementação da solução proposta, servindo como prova de conceito para o modelo.

No capítulo 5 são descritos os resultados do experimento proposto com o protótipo criado, analisando o sucesso da solução.

No capítulo 6 termina-se por sumarizar a solução e os resultados obtidos pelo sistema apresentado, além de indicar alternativas de trabalhos futuros.

2 FUNDAMENTAÇÃO

2.1 APRENDIZADO DE MÁQUINA

Por aprendizado de máquina entende-se a capacidade de um sistema computacional de aprender por observação de dados. Tal aprendizado costuma se basear em métodos estatísticos para melhorar uma aproximação ou minimizar uma função objetivo iterativamente, de forma a possibilitar um mecanismo de predição capaz de operar em dados ainda não observados.

2.1.1 APRENDIZADO SUPERVISIONADO

O método aprendizado pode ser inicialmente categorizado em supervisionado ou não-supervisionado, onde a diferença está na presença ou ausência da informação alvo de predição de forma explícita nos dados de treinamento. A nomenclatura usual para os dados de entrada do algoritmo varia entre *preditores* no meio estatístico ou *features* (características de entrada) no campo de reconhecimento de padrões, enquanto os dados de saída são chamados *respostas* ou *target variables* (variáveis alvo).

2.1.2 PROBLEMAS DE CLASSIFICAÇÃO

Outra distinção comum no meio do aprendizado estatístico é feita entre problemas de regressão e classificação. Enquanto o método de aprendizado é do tipo supervisionado como resultado da forma de coleta dos dados de treinamento, a natureza do dado podendo ser quantitativa ou qualitativa definirá o tipo de problema a ser estudado.

Variáveis alvo de natureza qualitativa são classes, também ditas categóricas ou discretas, e dão origem a problemas de classificação. Em diversos aspectos são semelhantes a problemas de regressão e alguns métodos podem ser utilizados em ambos os casos, especialmente em categorias ordenáveis - isto é, existe uma relação quantitativa entre as categorias.

Informações representadas em categorias são comumente codificadas em variáveis numéricas auxiliares, como 1 ou 0 para classes binárias (como vivo/morto ou presente/ausente). Classes com domínios mais abrangentes podem ser codificados por números naturais ou separadas em variáveis auxiliares (*dummy variables*), uma para cada classe indicando pertinência. Isto é, uma variável de K valores possíveis é transformada em K

variáveis binárias onde a k-ésima variável é 1 ou 0 se a classe observada é a de índice k ou não.

2.1.3 ÁRVORES DE DECISÃO

Um dos métodos de aprendizado estatístico mais frequentemente associados a processos decisórios é o baseado em árvores. Esse método consiste em iterativamente particionar o espaço de *features* e utilizar modelos super simples (como uma constante) em cada subdivisão.

Diversos métodos de construção dessa árvore de decisão foram propostos, sendo o mais comum e amplamente utilizado o método CART (*Classification And Regression Trees*). A princípio, o método poderia gerar árvores que sequencialmente dividem o espaço de *features* em um número qualquer de regiões, mas em geral a estratégia de divisões binárias é a de melhor resultado por generalizar divisões maiores e mitigar a ocorrência de regiões muito esparsas. Um exemplo de árvore CART pode ser visto na figura 2.1.

Dado um conjunto de N observações consistindo de p entradas e uma resposta alvo, denotaremos por (x_i, y_i) para $i = 1, ..., N$ os pares de *features* $x_i = (x_{i1}, ..., x_{ip})$ e variável alvo y_i. O algoritmo de construção da árvore deverá escolher as variáveis e respectivos pontos de corte que definirão a partição do espaço de entrada em M regiões $R_1, ..., R_M$.

Se cada região R_m define uma variável resposta c_m, podemos definir a saída da árvore como a função:

$$f(x) = \sum_{m=1}^{M} c_m I(x \in R_m) \tag{2.1}$$

Onde $I(\cdot)$ é a função indicadora, convertendo um predicado verdadeiro ou falso em 1 ou 0 respectivamente.

Para uma árvore T (inicialmente vazia), enumerando os $|T|$ nós terminais associados às regiões R_m cada uma com N_m observações, deseja-se então minimizar uma medida de impureza da partição m denotada por $Q_m(T)$.

De forma a sequencialmente construir a árvore, é utilizado um algoritmo guloso para iterativamente criar novas partições ao analisar a variável j no ponto s. Inicialmente, temos todo o conjunto de treinamento e uma partição genérica:

$$R_1(j, s) = \{X \mid X_j \leq s\} \qquad R_2(j, s) = \{X \mid X_j > s\} \tag{2.2}$$

Ao adicionar a partição de X por (j, s) à topologia da árvore T obtendo $T' \supset T$, deseja-se então resolver o problema de minimização:

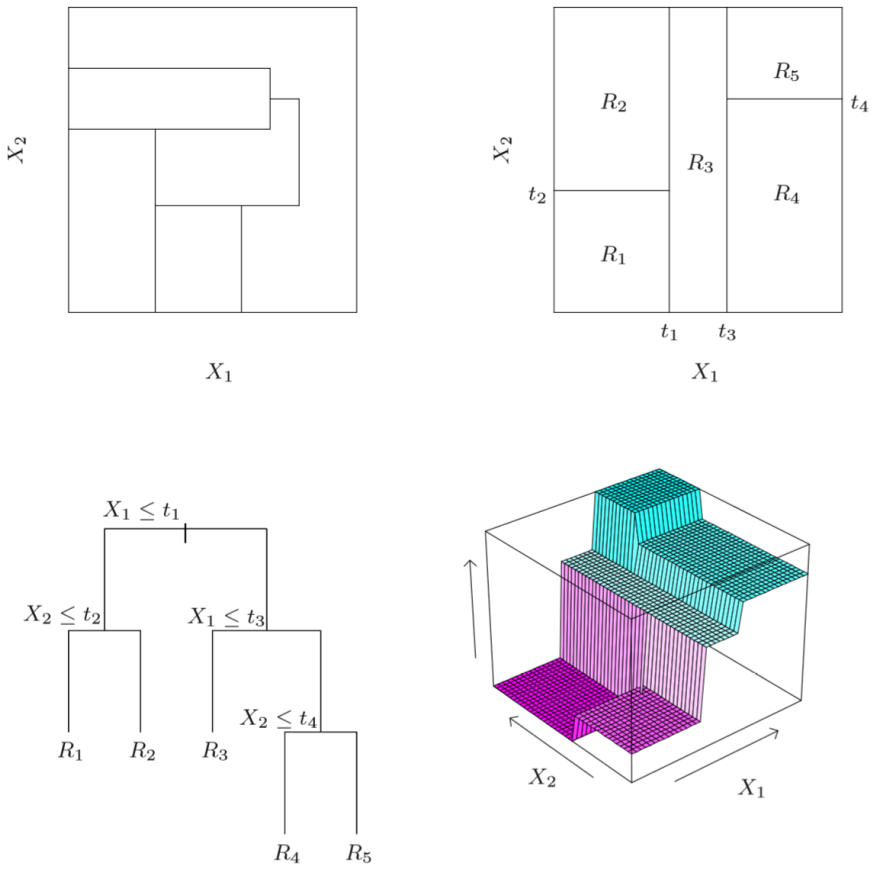

FIG. 2.1: Exemplo de resultado de modelagem CART. No canto superior esquerdo, uma partição genérica e complexa. No canto superior direito, uma sequência de partições binárias. Em seguida, a árvore de decisão associada e uma representação visual de superfície de predição. Fonte: Hastie et al. (2001)

$$\min_{j,s} N_1 Q_1(T') + N_2 Q_2(T') \tag{2.3}$$

E assim sucessivamente, até gerar a árvore final T_0. Neste momento é aplicado um algoritmo de poda da árvore de forma a reduzir a complexidade do modelo, efetivamente regularizando a ávore de decisão.

Para um parâmetro α de regularização, define-se o custo de complexidade da árvore T com a equação:

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha|T| \tag{2.4}$$

E assim deseja-se encontrar a árvore $T_\alpha \subset T_0$ que minimiza $C_\alpha(T)$. O parâmetro α regula a intensidade da regularização, controlando a troca entre complexidade e generalização do modelo. Múltiplos métodos para encontrar T_α são conhecidos e listados em Hastie et al. (2001).

O método apresentado funciona da mesma forma tanto para Árvores de Decisão aplicadas a problemas de regressão quanto a problemas de classificação, havendo diferença de aplicabilidade conforme a escolha da medida de impureza $Q_m(T)$, especificada a seguir.

Define-se primeiramente a proporção da classe k na partição m por:

$$p_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k) \tag{2.5}$$

e a predição das observações na partição m por:

$$\kappa(m) = \arg\max_k p_{mk} \tag{2.6}$$

Finalmente, para problemas de classificação as três medidas clássicas de impureza de uma partição são:

$$\text{Missclassification Error:} \quad \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i \neq k(m)) = 1 - p_{m\kappa(m)} \tag{2.7}$$

$$\text{Gini Index:} \quad \sum_{k \neq k'} p_{mk} p_{mk'} = \sum_{k=1}^{K} p_{mk}(1 - p_{mk}) \tag{2.8}$$

$$\text{Cross-Entropy:} \quad -\sum_{k=1}^{K} p_{mk} \log p_{mk} \tag{2.9}$$

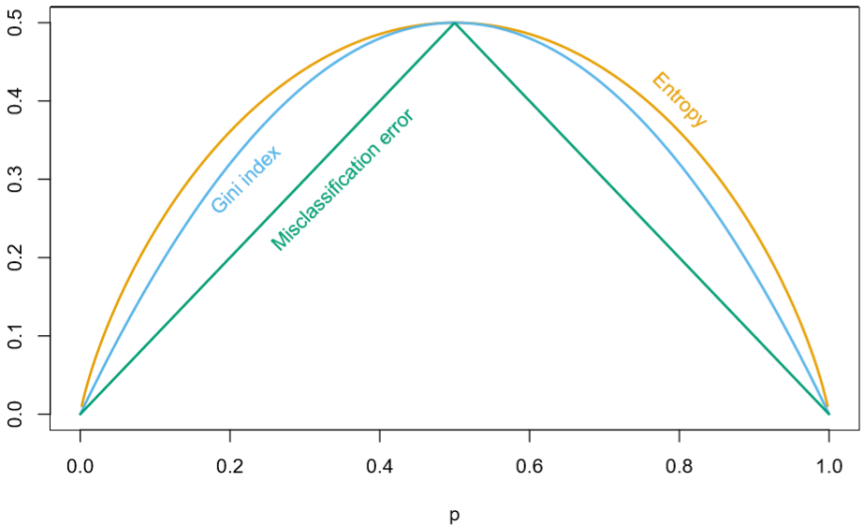

FIG. 2.2: Métricas de impureza (escaladas para comparação) em função da proporção p em um caso de duas classes. Fonte: Hastie et al. (2001)

O *Gini Index* pode ser entendido como uma medida proporcional ao erro da atribuição de probabilidades de classificação ou como o agregado das variâncias da classificação de cada classe. É o critério padrão utilizado pela biblioteca *scikit-learn*.

As três métricas de impureza se comportam de forma similar, porém o *Gini Index* e a *cross-entropy* são diferenciáveis e portanto mais amigáveis a otimizações numéricas. A diferença de suavidade das funções pode ser vista na figura 2.2. Outras características como maior sensibilidade a mudanças de probabilidades fazem com que estes sejam os mais recomendados para a etapa de criação da árvore, enquanto o *Misclassification Error* é usado na etapa de poda (HASTIE et al., 2001).

2.1.4 VALIDAÇÃO CRUZADA

A validação é uma técnica que visa tornar a tática escolhida em ótima, assim sendo a validação é principalmente um meio de garantir que a abordagem escolhida para o problema esteja tão próxima da realidade quanto for possível, neste caso o conjunto de amostras é tratado para aprimorar a aplicação da tática, o tratamento pode funcionar de maneiras de ferentes de uma tática para outra, por fim a tática de validação mais utilizada e precisa quanto a validar o modelo é a validação cruzada, a qual apesar da eficiência e precisão pode não ser utilizada em detrimento de outras táticas por ser muito custosa em termos computacionais.

A validação cruzada se baseia em redistribuir a amostra de teste em diversas sub-

amostras de treino para então escolher-se recursivamente tais conjuntos de amostras e usar o restante como amostra de treino esta abordagem visa estimar o valor do E_{out} que não pode ser calculado sem treino ou verificação do modelo já em uso, então calculam-se os erros para assim se escolher como amostra de teste a amostra, a qual contenha os melhores índices de erro; porém, como visto previamente, forçar os valores de erro na amostra e fora dela para minimiza-los pode acarretar em problemas de overfitting e neste caso os erros calculados durante a validação iriam destoar drasticamente do seu valor real.

Outra técnica de validação amplamente utilizada é a reamostragem, nesse caso o modelo escolhido, mesmo após processos de verificação, não atende as expectativas quando implementado e por isso se necessita refazer o treinamento, assim sendo se colhem novas amostras para refazer o processo de treino, é importante que não se usem as amostras já colhidas, pois estas iriam moldar o novo modelo com certo juízo de valor, contribuindo fatalmente para o problema de overfitting.

Outra forma de validar um modelo é pela verificação continua durante o treino, neste caso a cada iteração do treinamento calculam-se os erros do modelo em relação ao esperado para assim melhorar o modelo com alterações pontuais, similar ao processo de entrega de um produto em que a cada versão busca-se melhora-lo e consertar os problemas descobertos, assim como a tática de validação cruzada esse método é muito custoso computacionalmente e por isso pode ser desestimulado em relação aos outros.

2.2 CARACTERÍSTICAS DE DIGITAÇÃO

2.2.1 EVENTOS DE TECLA

A amostra de uma dinâmica de digitação consistem em uma sequência de eventos de tecla $a = (t_a^{\downarrow}, t_a^{\uparrow}, C_a)$ onde $t_a^{\downarrow}$ é o instante de tempo em que a tecla foi pressionada, $t_a^{\uparrow}$ o instante de tempo em que a pressão foi removida (relaxada) e C_a o caractere sinalizado.

Aqui consideramos que o evento de tecla $a = (t_a^{\uparrow}, t_a^{\downarrow}, C)$ difere do evento $b = (t_b^{\uparrow}, t_b^{\downarrow}, C)$ mesmo que ambos correspondam ao mesmo caractere da entrada C, pois a informação temporal está embutida no evento de tecla.

Como cada tecla está associada a dois instantes de tempo (pressionar e relaxar), a noção de teclas *consecutivas* precisa ser bem definida: consideramos os eventos de tecla a e b consecutivos se nenhum outro evento de tecla c é tal que $t_a^{\downarrow} < t_c^{\downarrow} < t_b^{\downarrow}$. Ou seja, intuitivamente teclas consecutivas foram pressionadas de forma consecutiva independente dos tempos em que foram relaxadas.

As características de dinâmica de digitação utilizadas são obtidas através de esta-

tísticas após múltiplas amostragens do tempo entre eventos de teclas. Especificamente, definiremos eventos artificiais $\alpha = (t_\alpha, S_\alpha)$ obtidos ao combinar os eventos de tecla de forma a gerar um evento com uma única variável temporal e associado a uma sequência de caracteres de tamanho qualquer.

2.2.2 EVENTOS DERIVADOS

Definimos o evento de *down-down* a partir dos eventos de tecla a e b por $DD.a.b = (t_{DD.a.b}, S_{DD.a.b})$, onde $S_{DD.a.b} = C_aC_b$ (isto é, o digrama formado pelos caracteres dos eventos de tecla a e b) e $t_{DD.a.b}$ especificado pela equação 2.10.

$$t_{DD.a.b} = t_b^{\downarrow} - t_a^{\downarrow} \tag{2.10}$$

De forma explícita, podemos dizer que o tempo do evento $DD.a.b$ é a diferença entre o tempo de pressionar no evento de tecla a e o tempo de pressionar no evento de tecla b.

De forma análoga, definimos o evento de *up-down* a partir dos eventos de tecla a e b por $UD.a.b = (t_{UD.a.b}, S_{UD.a.b})$, onde $S_{UD.a.b} = C_aC_b$ e e $t_{DD.a.b}$ especificado pela equação 2.11.

$$t_{UD.a.b} = t_b^{\downarrow} - t_a^{\uparrow} \tag{2.11}$$

Ou seja, o tempo do evento $UD.a.b$ é a diferença entre o tempo de relaxar do evento de tecla a e o tempo de pressionar do evento de tecla b. Diferentemente de $DD.a.b$, é possível que $UD.a.b$ possua valores de tempo negativos, já que o evento de tecla b pode ser consecutivo do evento de tecla a e mesmo assim ter seu instante de tecla pressionada anterior ao instante do evento de tecla a ter sido relaxada.

De forma complementar, definimos o evento de *hold* a partir do evento de tecla a como $H.a = (t_{H.a}, S_{H.a})$, onde $S_{H.a} = C_a$ e $t_{H.a}$ é dado pela equação 2.12.

$$t_{H.a} = t_a^{\uparrow} - t_a^{\downarrow} \tag{2.12}$$

$H.a$ é interpretado como o evento da tecla a permanecer pressionada. Como estaremos particularmente interessados em derivar o evento *down-down*, *up-down* e *hold* a partir de eventos de tecla consecutivos, podemos derivar a seguinte relações a partir das equações 2.10, 2.11 e 2.12:

$$t_{H.a} + t_{UD.a.b} = t_{DD.a.b} \tag{2.13}$$

Após derivar os eventos *down-down*, *up-down* e *hold* para cada digrama da sequência (isto é, pares de eventos de tecla consecutivos) podemos analisar esta nova sequência de eventos derivados e calcular estatísticas simples como médias e desvios padrão.

Considerando que cada sequência de caracteres S está associado a N_S^{DD} eventos *down-down*, N_S^{UD} eventos *up-down* e N_S^H eventos *hold*, definimos para os caracteres x e y:

$$T_{x.y}^{DD} = \frac{1}{N_{xy}^{DD}} \sum_{S_{DD.a.b}=xy} t_{DD.a.b} \tag{2.14}$$

$$T_{x.y}^{UD} = \frac{1}{N_{xy}^{UD}} \sum_{S_{UD.a.b}=xy} t_{DD.a.b} \tag{2.15}$$

$$T_{x}^{H} = \frac{1}{N_{x}^{H}} \sum_{S_{H.a}=xy} t_{DD.a.b} \tag{2.16}$$

Isto é, as equações 2.14 e 2.15 definem respectivamente os intervalos médios de entre o pressionar de teclas e o intervalo médio entre o relaxar da primeira e o pressionar da segunda nos digramas consecutivos, enquanto a equação 2.16 define o tempo médio de pressão.

As equações 2.14, 2.15 e 2.16 definem as *features* que serão utilizadas neste trabalho. Por motivo de simplicidade de modelo e pelo crescimento quadrático no número de *features* conforme o número de caracteres analisados, foi escolhido não fazer uso dos desvios padrões correspondentes.

De forma geral, poderia se guardar todas as sequências dos eventos (H,UD,DD) para cada digrama da amostra. Assim é possível reconstruir a sequência e posteriormente calcular outras *features* (além dos tempos médios) para refinar os modelos.

Entretanto, um dos objetivos da solução a ser apresentada é ser de uso genérico e potencialmente utilizada como fortalecimento de senhas. Isto é, o campo digitado e analisado pode ser a sequência de caracteres que compõe a senha do usuário e estaria sendo guardada de forma não criptografada em um serviço terceiro.

Portanto, é necessário sacrificar o poder de reprocessamento e armazenar as estatísticas já agregadas, ainda correndo o risco de deixar transparecer certos aspectos distribucionais da composição de caracteres das senhas dos usuários.

3 TRABALHOS RELACIONADOS

3.1 TRABALHOS ACADÊMICOS

Monrose et al. (1999) inicialmente propôs que a análise de padrões na dinâmica de digitação como parte da verificação de uma senha. O algoritmo sugerido portanto era equivalente a o uso de uma senha fortalecida onde não somente a sequência de caracteres precista estar correta, mas como sua inputação precisa ser reconhecida. A solução chegou a ser comercializada, porém os autores admitiram que senhas curtas e padronizadas não capturam padrões de digitação suficiente e a performance do sistema chegava a taxas de 40% de falsos negativos.

Bergadano et al. (2002) utilizou textos maiores (680 caracteres) e não analisava diretamente os tempos dos eventos de teclas. Ao invés disso, utilizaram os eventos ordenadamente para calcular uma métrica de informação chamada Grau de Desordem (*Degree of Disorder*) para comparar duas sequências de trigramas (trios de teclas) ordenados. Esta medida teve sucesso em reduzir o efeito da variância de amostragem temporal, pois analisa apenas a ordem relativa de velocidade dos trigramas e não os tempos absolutos.

Lau et al. (2004) aprofundou a comparação entre métodos tradicionais de análise temporal como médias e desvio padrão com o uso do Grau de Desordem, concluindo que este era consideravelmente mais estável e adequado para autenticação por dinâmica de digitação.

Gunetti e Picardi (2005) expandiu a análise de Grau de Desordem ao analisar não apenas trigramas, mas para todos os n-gramas. Os resultados experimentais alcançaram menos de 5% de falso positivos e 0.005% falsos negativos, porém a solução não era escalável. O Grau de Desordem é na verdade uma métrica de distância entre amostragens, então para cada entrada é necessário o cálculo de distância para todas as amostras de treinamento de todos os usuários.

Hu et al. (2008) obteve sucesso em aumentar a performance do processo de autenticação em 66.7% ao utilizar um algoritmo de clusterização para limitar o conjunto de amostras a serem comparadas, mantendo as taxas de falsos positivos e falsos negativos.

Outros trabalhos propuseram novas métricas e padrões a serem analisados, enquanto trabalhos recentes discutem abordagens por parte do algoritmo de verificação. O aumento da complexidade dos modelos de verificação, especialmente através de novos algoritmos

baseados em aprendizado de máquina, reintroduziram o uso da análise temporal e métodos associados para lidar com sua variância.

Morales et al. (2015) mostrou um aumento significativo da consistência e aplicabilidade das séries temporais ao aplicar uma normalização localizada, isto é, entre amostras de um mesmo usuário ao invés de aplicada ao banco de amostras inteiro.

Haque et al. (2016) apresenta uma solução baseada somente nos métodos estatísticos dos tempos dos eventos e alcança 93% de acurácia (5% de falsos positivos e 8% de falsos negativos).

Neste trabalho foram utilizados os métodos estatísticos semelhantes a Haque et al. (2016) e Monrose et al. (1999), porém com ênfase na estrutura do sistema como um serviço externo ao sistema que deseja efetuar a autenticação.

3.2 FERRAMENTAS DE MERCADO

Duas soluções no mercado apresentam grande interseção de funcionalidades com a solução proposta. O serviço provido pela TypingDNA se baseia no mesmo mecanismo de uma classe JavaScript injetada no código da página do cliente que efetua a coleta dos padrões de digitação.

Diferentemente deste trabalho, a solução da TypingDNA não faz uso de digramas ou quaisquer outra sequência com mais de um caractere. De acordo com a documentação, esta decisão foi tomada pois digramas permitiriam a reconstrução da senha original (TYPINGDNA).

Ao utilizar apenas caracteres isolados, a TypingDNA coleta dados de 44 teclas (não especificadas) que geram um vetor de 320 *features*, consideravelmente mais do que neste trabalho (150). Considerando que apenas são utilizados 1-gramas, isto significa mais de 7 *features* para cada caractere (TYPINGDNA).

Outra solução no mercado encontrada é a KeyTrac, funcionando de forma equivalente ao TypingDNA, porém com duas APIs distintas: uma para ser usada no campo de senha e outra em campos de texto livre. Menos detalhes são dados sobre que *features* são disponibilizados, porém a KeyTrac menciona o uso de uma codificação que obscura a amostragem para aumentar a segurança do serviço (KEYTRAC). O diagrama de interação dos componentes apresentada pela KeyTrac está representada da figura 3.1.

Ambos os serviços retornam na API uma métrica de certeza que pode ser utilizada pelo usuário para decidir a verificação. Porém, nenhum dos dois serviços provê formas de configuração que permitam ao cliente escolher exatamente o modelo e as *features*

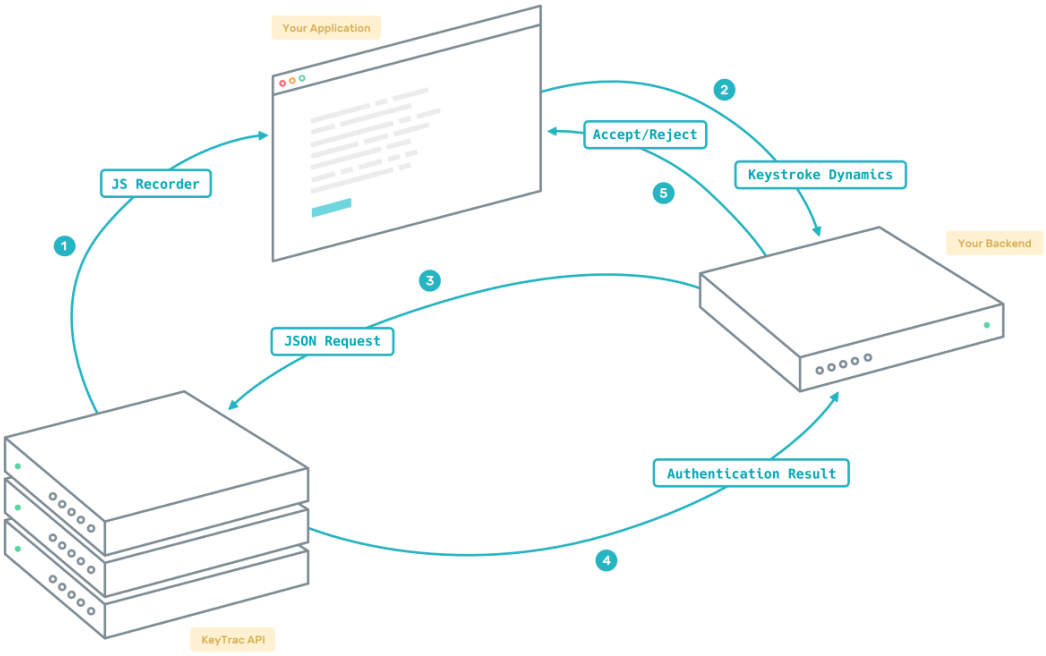

FIG. 3.1: Diagrama de interação entre serviços no serviço KeyTrac. Fonte: KeyTrac

utilizadas. Embora nenhum dos dois mencione sua tecnologia interna, essa uniformidade de interface indica que eles utilizam um mesmo modelo para autenticar múltiplos usuários (possivelmente todos).

Neste trabalho nós deixamos especificada apenas a comunicação de alto nível entre o serviço de autenticação e o serviço cliente. Deste modo, no protótipo apresentado pudemos utilizar o serviço para treinar um algoritmo separado para cada usuário, de forma que cada pessoa esteja associada a um modelo de reconhecimento dedicado a reconhecer seu padrão de digitação.

4 SOLUÇÃO PROPOSTA

4.1 DESCRIÇÃO CONCEITUAL

O sistema de autenticação proposto consiste em um serviço que ofereça as operações básicas de manutenção de credenciais para usuários de clientes mais uma operação de verificação. Essas funcionalidades são acessadas via API REST, a critério do sistema que requer autenticação de seus usuários.

Como a solução proposta é um serviço de autenticação para usuários de outros serviços, é necessário identificar o cliente que está fazendo a requisição para que seja possível segmentar corretamente que usuários lhe pertencem. Tal segmentação costuma ser realizada via uma espécie de *token*, que também define a forma de cobrança pelo uso do serviço (em geral, número de requisições por mês associadas ao *token*).

A entidade administrada via operações de CRUD é a do usuário, onde em particular interessam as operações que modificam suas credenciais. Na solução proposta, a credencial é na verdade um conjunto de amostras de padrões de digitação mais o modelo resultado do treinamento de algoritmo de aprendizado de máquina para classificação para reconhecer amostras semelhantes. A relação entre as entidades do sistema pode ser vista no diagrama da figura 4.1.

Na figura 4.2 pode ser visto o fluxo de criação de uma credencial (ou de forma análoga, uma reamostragem do mesmo usuário). O cliente faz uma requisição enviando os dados coletados sobre a dinâmica de digitação (utilizando a biblioteca auxiliar) para a API, que irá extrair as *features*, persistir a amostra (exceto quando configurado para não o fazer, como recomendado em caso de dado sensível) e as *features* extraídas, criar o modelo associado ao usuário e salvá-lo.

A sincronicidade de eventos pode ser ajustada conforme a demanda do sistema. A criação do modelo pode ser síncrona e imediata, ou apenas um gatilho de agendamento de tarefas (nesse caso, salvar o modelo não estaria presente no diagrama). Caso a criação do modelo seja de forma síncrona, como é o caso da implementação no protótipo apresentado, o recomendado é que a requisição feita pelo cliente seja assíncrona por causa da latência associada ao treinamento do modelo (nesse caso, cuidados extras devem ser tomados ao tentar utilizar uma credencial existente).

Na figura 4.3 está diagramado o fluxo de autenticação. Assim como durante a cri-

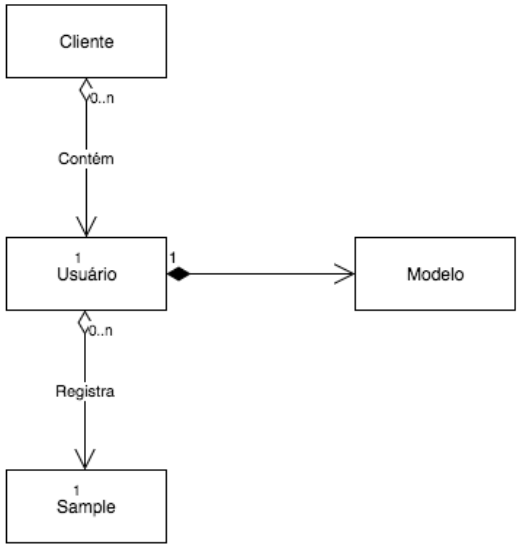

FIG. 4.1: Diagrama de classes do sistema.

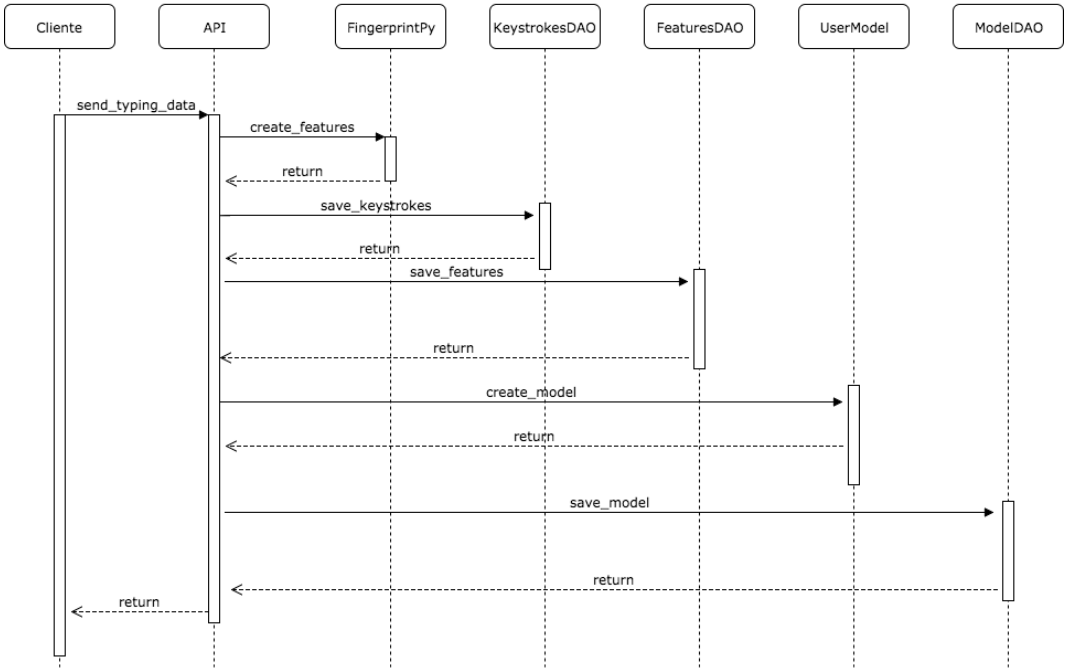

FIG. 4.2: Diagrama de sequência da criação uma credencial.

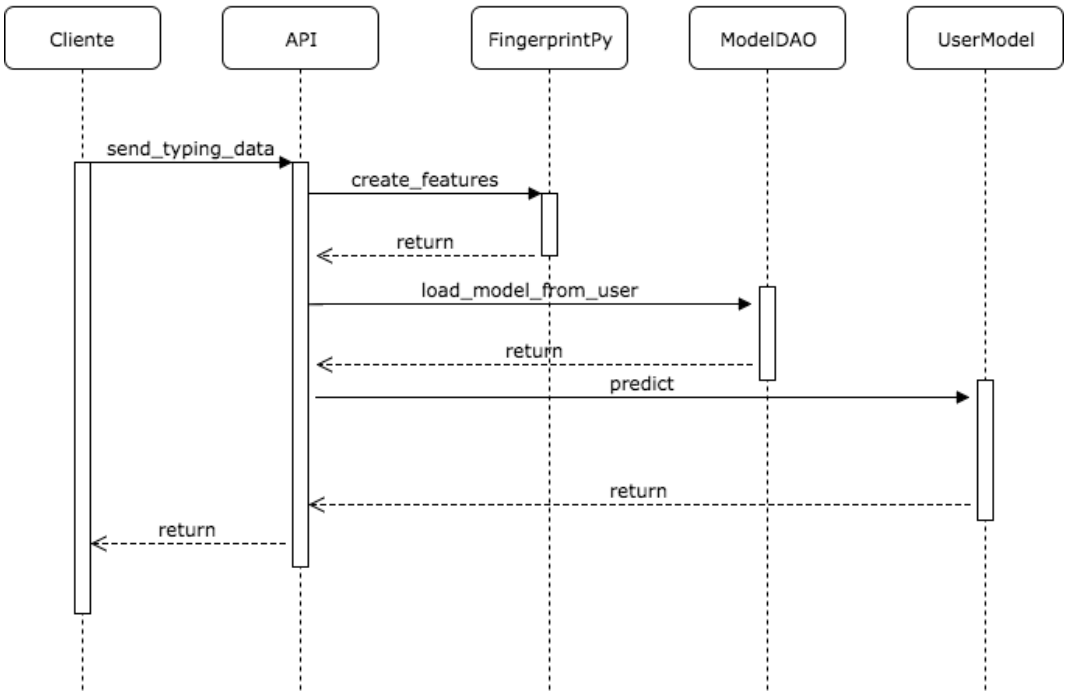

FIG. 4.3: Diagrama de sequência do processo de autenticação.

ação/manutenção de credenciais, o usuário faz uma requisição enviando a amostra de dinâmica de digitação e a API aplica a transformação para o espaço de *features*. Desta vez, basta carregar o modelo associado ao usuário do banco de dados e retornar a predição de verificação positiva.

Ao invés de simplesmente retornar a classe de saída do preditor, utilizamos modelos que também tenham como saída um grau de confiança da predição. Assim, o que de fato é retornado é um indicador (no intervalo $[0, 1]$) de aceitação da identidade.

Como utilizar a aceitação do padrão digitação como critério de autenticação fica a critério do cliente, que pode admitir diferentes níveis de tolerância ou diferentes ações de resposta a uma verificação negativa.

Ao observar que a extração de *features* é feita no *backend* da API e um modelo é carregado individualmente para cada requisição, este *design* permite uma gama maior de opções de configuração por parte do cliente como que tipo de modelo deseja treinar, que tipo de *features* extrair e se a amostra original deve ser armazenada. Tal flexibilidade não está presente nas soluções concorrentes descritas na seção 3.2.

Para auxiliar o uso da API, a biblioteca desenvolvida automatiza a coleta de dados de dinâmica de digitação nos campos definidos pelo cliente em seu site, além de disponibilizar métodos de tratamento de dados coletados como truncamento e sub-amostragem.

A figura 4.4 apresenta um exemplo de sistema conjunto mostrando a interação entre

o cliente e o serviço de autenticação, semelhante ao observado na figura 3.1. Entretanto, esta estrutura não é obrigatória: o cliente pode enviar as requisições diretamente do JavaScript da página, por exemplo, eliminando a necessidade de comunicação entre a API e o backend. Neste modelo, a API de dinâmica de digitação serve de forma semelhante a um validador de formulário superficial.

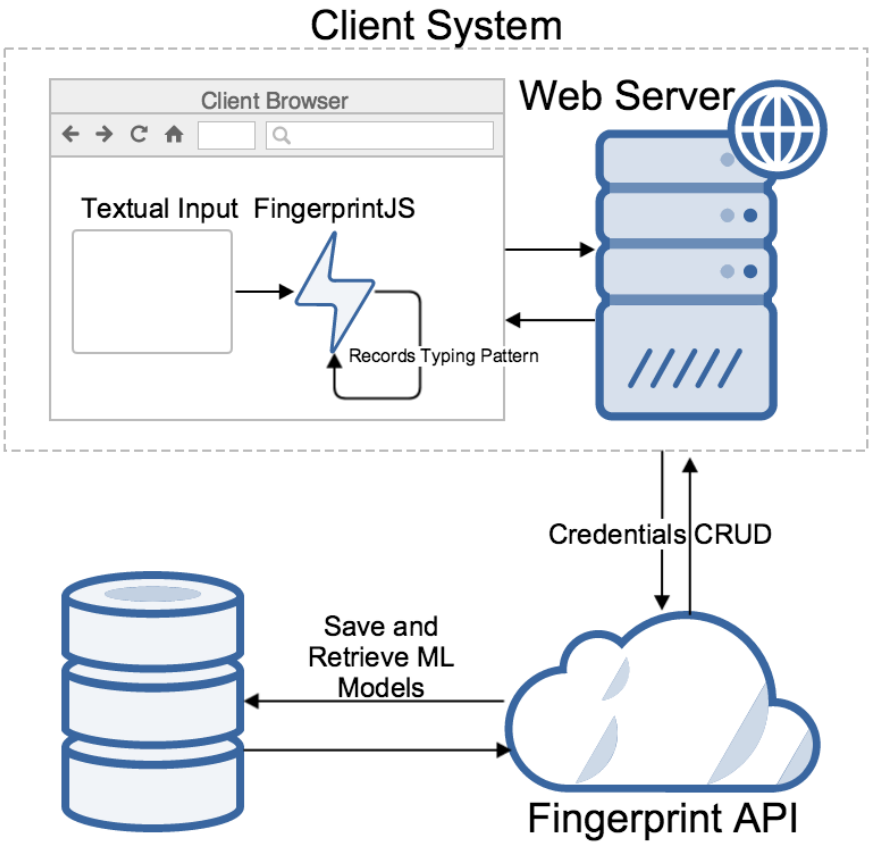

FIG. 4.4: Visão geral dos componentes do protótipo.

4.2 PROTÓTIPO

Embora o serviço consista de uma API - podendo ser, portanto, implantado como um único servidor posteriormente escalado por replicação, desejou-se definir também a infraestrutura de processamento de dados. Para o funcionamento do serviço é necessário que modelos preditivos sejam constantemente retreinados de forma automatizada.

Como a carga de um sistema de processamento offline é extremamente variável, buscamos uma arquitetura de servidores e serviços que seja escalável e flexível com baixo custo. O provedor de computação em nuvem escolhido foi a Amazon Web Services devido à sua grande flexibilidade e baixo custo.

O modelo esquemático do sistema de processamento está representado na figura 4.5. De fato, a arquitetura apresentada é genérica e totalmente reprodutível para qualquer sistema de processamento de dados em modo *offline batch*, isto é, quando não há a necessidade de processamento instantâneo.

A arquitetura do sistema de processamento é capaz de concentrar não somente as tarefas de treinamento de modelos (quando são cadastradas novas amostras, novos usuários ou o modelo é atualizado), mas também rotinas de ETL que porventura sejam necessárias quando a mudança de modelo exija uma mudança de extração de *features*. Desta forma, o sistema comporta a eventual ocorrência de migração de esquemas com simplicidade.

Não somente no contexto do protótipo apresentado, mas em qualquer sistema com demanda de processamento de dados, a infraestrutura apresentada possui baixo custo e extrema facilidade de implantação. A especificação da infraestrutura na AWS também foi disponibilizada como código aberto (CHAVES, 2017).

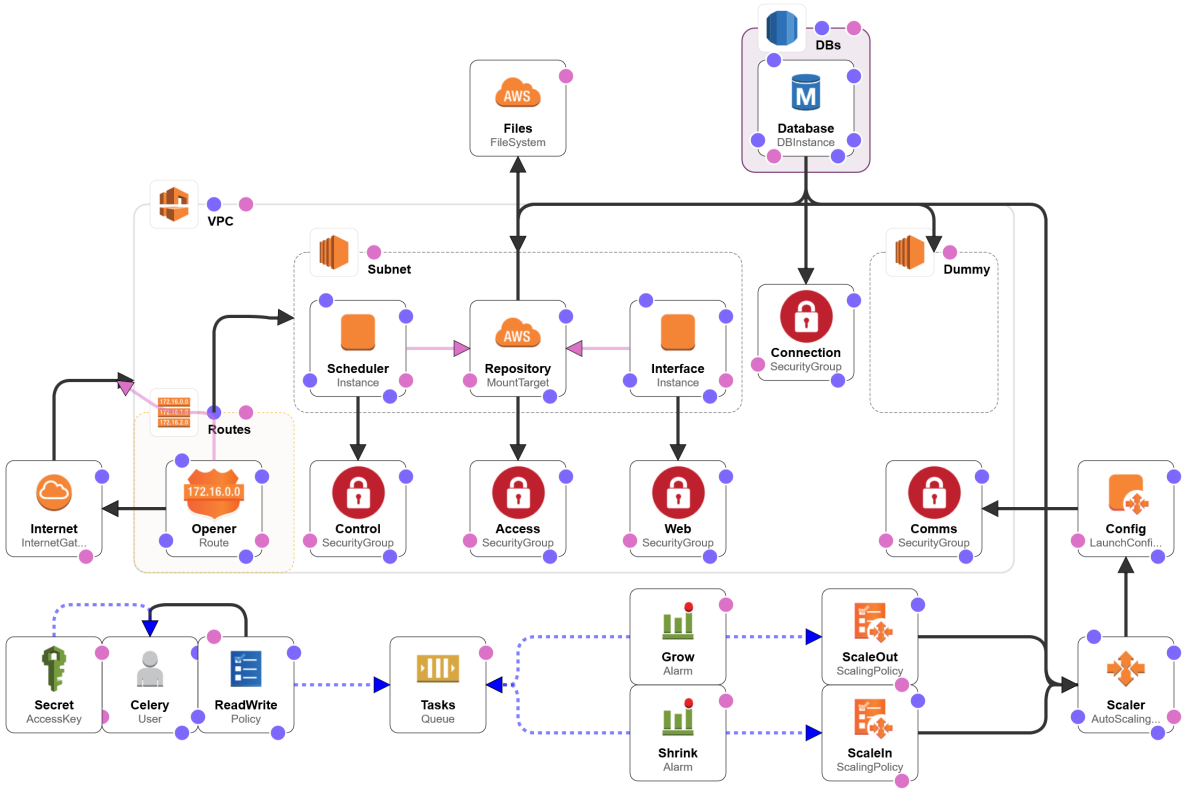

FIG. 4.5: Diagrama de serviços no *Cloud Formation*, na notação padrão de serviços AWS.

4.2.1 REDES

No modelo de nuvem da Amazon Web Services (AWS), as máquinas virtuais precisam estar dentro de uma Rede Virtual Privada (**VPC** - Virtual Private Network), que é

dividida em Sub-redes (**Subnet**) especificadas em diferentes Zonas de Disponibilidade (Availability Zones, representam os diferentes servidores da AWS pelo mundo).

Para dar acesso à Internet para as máquinas, é necessário associar um Gateway de Internet (*Internet* - Internet Gateway) à **VPC** e uma tabela de rotas de roteamento (**Routes**) contendo uma rota de abertura (**Opener** - Route) para a **Subnet** para redirecionar o tráfego ao endereço local do gateway.

4.2.2 MÁQUINAS VIRTUAIS

Duas máquinas virtuais persistentes são utilizadas para o sistema de processamento. Uma delas agindo como servidor *web* para a interface gráfica de controle e monitoramento do sistema (**Interface**), enquanto a outra contém o serviço de agendamento de tarefas, agindo como nó mestre do cluster de processamento (**Scheduler**).

Para facilitar a coordenação da interface, do agendador e dos nós de processamento, todas as máquinas compartilham dados por um volume montado em comum em seus sistemas de arquivos (**Repository** - Mount Target). Esse volume montado está associado a um Sistema de Arquivos Elástico (**Files** - Elastic File System). É também nesse sistema de arquivos compartilhados que são salvos os *logs* do sistema de processamento, de forma centralizada.

4.2.3 DADOS

Além do sistema de arquivos, é necessário um banco de dados para persistir as informações relativas ao agendamento de tarefas além dos resultados do processamento (**Database** - DbInstance). Em particular, escolhemos utilizar o PostgreSQL como SGBD para os dados relacionados à API e para os dados de análise.

Ainda, para poder coordenar os nós de processamento, o **Scheduler** precisará de um mecanismo de comunicação distribuída. Para isso usa-se um serviço de fila de mensagens (**Tasks** - Queue). O sistema de escalonamento de nós processadores irá utilizar esta fila como parâmetro de decisão: uma fila muito grande de tarefas deverá indicar a necessidade de instanciar mais uma máquina de processamento.

4.2.4 PROCESSAMENTO

As máquinas de processamento serão criadas com base nos gatilhos (**Srhink** e **Grow** - Alarm), que a cada minuto irão verificar o tamanho da fila **Tasks**. Ao sair dos limites estipulados, os alarmes irão acionar respectivamente as políticas de escalonamento (**ScaleOut**

e **ScaleIn** - ScalingPolicy) que alocam e desalocam uma máquina de processamento.

O sistema de escalonamento (**Scaler** - AutoScalingGroup) irá instanciar as máquinas dentro da **Subnet** com as especificações de trabalho semelhantes ao nó mestre **Scheduler**, consumindo porém máquinas virtuais do banco de máquinas leiloadas sob demanda (Spot Instances) com menor custo.

5 ESTUDO DE CASO

Para confirmar a eficácia de utilizar um modelo de aprendizado de máquina treinado exclusivamente para cara usuário, foi utilizado um *dataset* de dinâmica de digitação público (licenciado via *Creative Commons*) e disponibilizado por Suman (2017). Neste experimento, foi treinada uma árvore de decisão para cada usuário, com o cuidado em separar os conjuntos de treinamento e testes na proporção 80/20 e de forma a preservar aproximadamente a proporção de amostras de cada usuário em cada conjunto - isto é, um quinto das amostras de todo usuário foi incluída no conjunto de testes.

Neste experimento, visto na figura 5.1, obtivemos uma média de falsos positivos de 17.94% e falsos negativos em 0.47%. Tais valores estão compatíveis com os trabalhos relacionados (como Monrose et al. (1999) e Bergadano et al. (2002)), mesmo utilizando apenas estatísticas básicas e digramas como fontes de *features*.

Para provar a usabilidade do sistema e eficiência da solução proposta, tal como a plausibilidade de treinar um modelo para cada usuário, uma prova de conceito foi desenvolvida com uma aplicação cliente para ser integrada com a solução desenvolvida como protótipo. As figuras 5.2 e 5.3 mostram as telas e os textos utilizados no protótipo. Os parâmetros utilizados para o treinamento da árvore de decisão não foram otimizados, mantendo as recomendações padrão da biblioteca de aprendizado de máquina utilizada (PEDREGOSA et al., 2011). Todos os componentes foram desenvolvidos em código aberto de forma a facilitar a reprodução dos resultados (MEIRELES, 2017).

O objetivo da prova de conceito era observar a usabilidade da aplicação, facilidade de integração com a aplicação cliente, desempenho computacional e desempenho do algoritmo de aprendizado de máquina. O sistema foi testado com 8 usuários que não reportaram queda de usabilidade. O *dataset* público mostrou que a abordagem teórica escolhida apresenta resultados semelhantes a trabalhos relacionados e o estudo de caso com o protótipo mostrou que o *design* da biblioteca auxiliar e da API permitem o desenvolvimento de um serviço de usabilidade indistinguível.

Do ponto de vista de infraestrutura, o sistema de processamento respondeu de forma esperada, porém sub-ótima. Se por um lado rodar o experimento em um Macbook Pro de configurações mínimas custava 15 minutos, o mesmo experimento controlado pelo software de agendamento de tarefas chegava a alcançar 1 hora. Essa deterioração se deve principalmente ao imenso *overhead* de orquestração de tarefas assíncronas entre 10

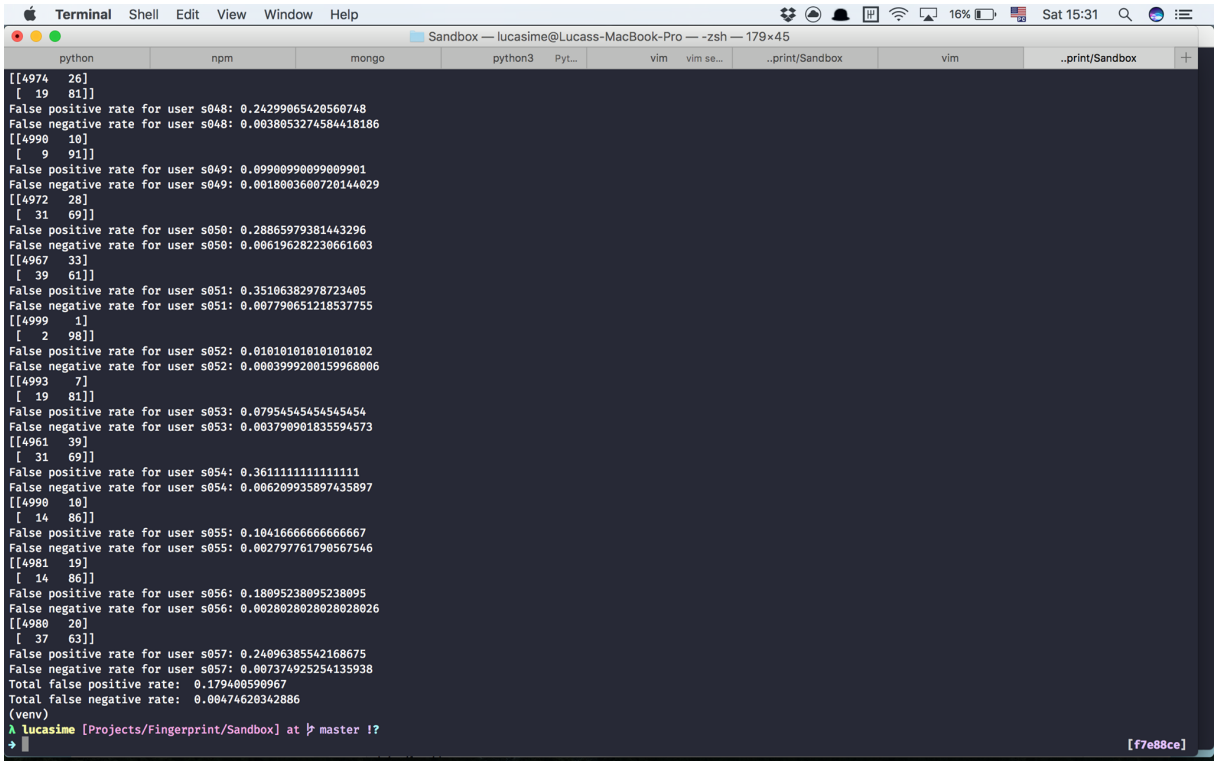

FIG. 5.1: Captura de tela do experimento com o *dataset* público.

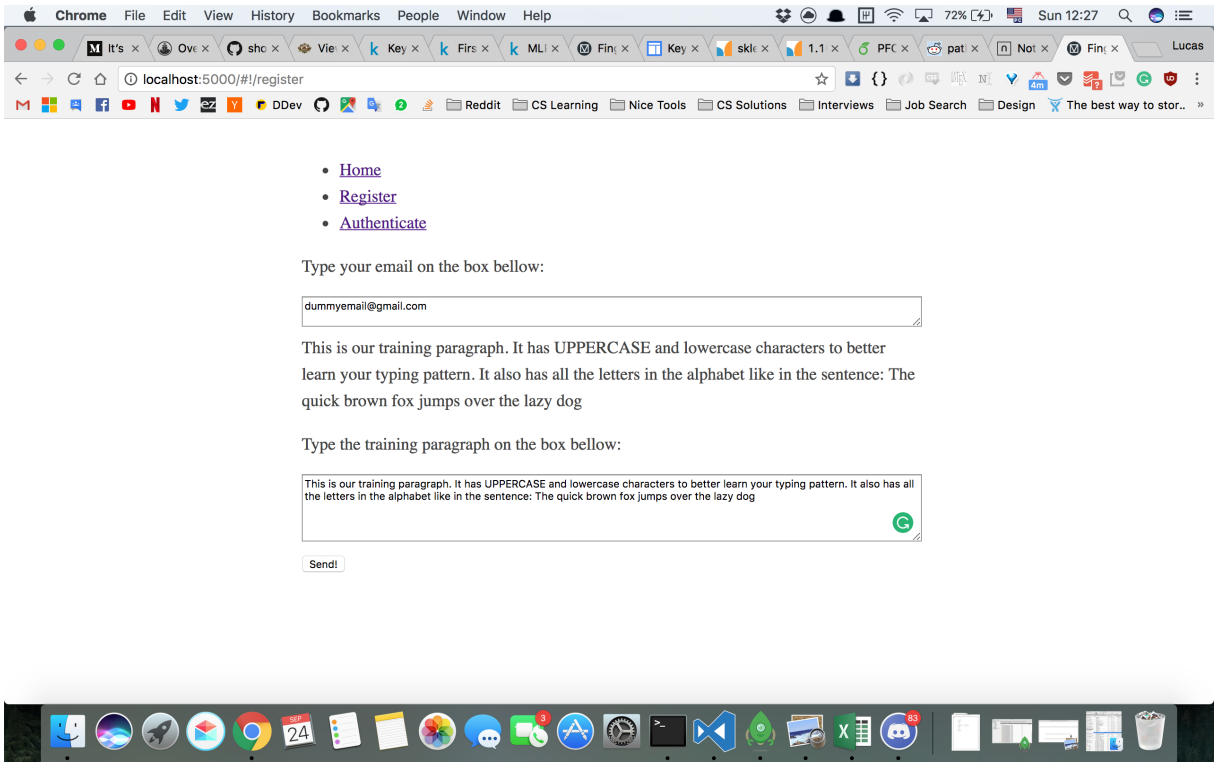

FIG. 5.2: Captura de tela com o formulário de registro e amostragem do protótipo.

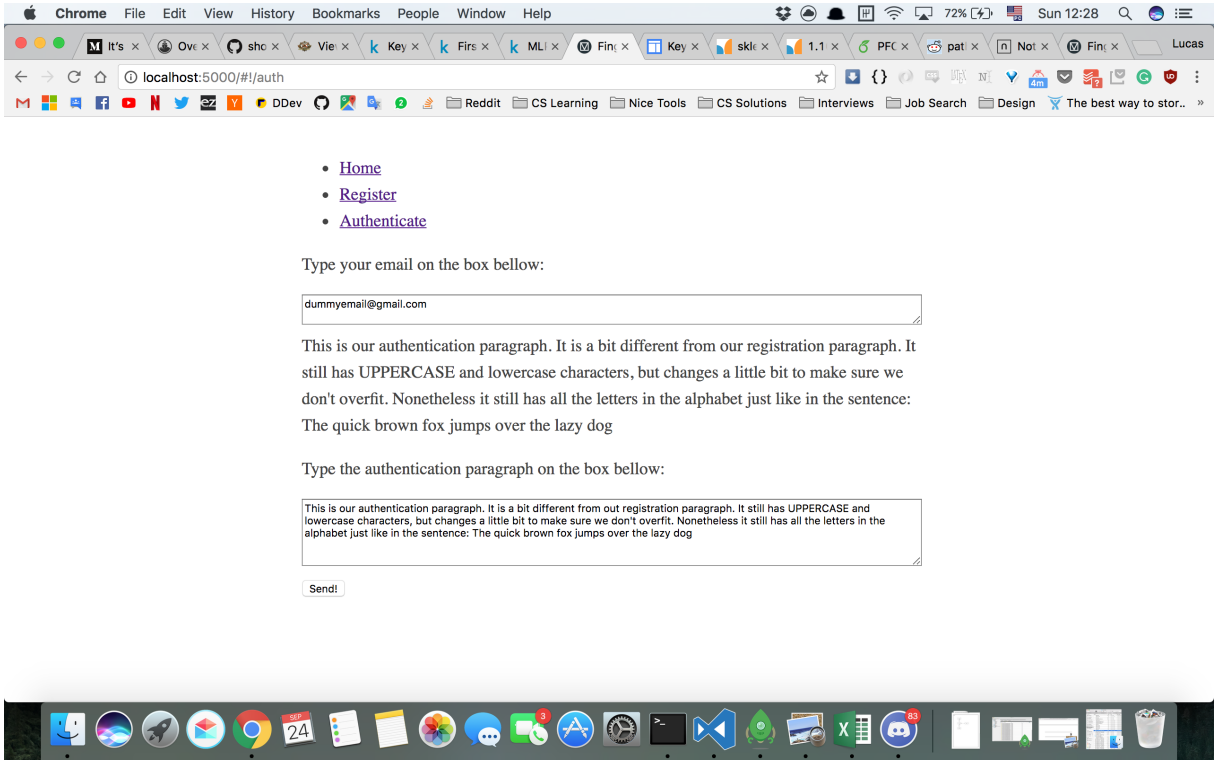

FIG. 5.3: Captura de tela com o formulário de verificação de identidade do protótipo.

nós processadores e dois nós centrais, de forma que os custos de coordenação e troca de mensagens não compensam para se instanciar dez máquinas de 1 núcleo de CPU e 1 GB de memória RAM.

Ainda assim, o sistema de processamento funcionou conforme o esperado e cumpre seu objetivo de permitir observabilidade e auditoria dos procedimentos de cálculo *offline*, além de servir de suporte para futuras operações relacionadas a governança de dados.

6 CONCLUSÃO

A análise de padrões na dinâmica de digitação de usuários foi proposta como uma possível direção para resolver diversos problemas relacionados à autenticação de usuários em sistemas computacionais, porém o mercado ainda carece de soluções comerciais flexíveis diante a variedade de necessidades do mercado.

Foi apresentado um modelo de sistema de autenticação baseado em dinâmica de digitação que pudesse ser configurado para abranger casos de uso de autenticação de senhas ou textos livres, mantendo-se transparente ao usuário e de fácil implantação pelo cliente como um serviço externo exposto via API REST.

Um protótipo foi elaborado para exemplificar o uso do sistema, utilizando uma modelagem simplificada dos métodos de reconhecimento de padrão de digitação como prova de conceito. O modelo simplificado teve sua efetividade confirmada com um *dataset* público, enquanto a usabilidade do protótipo foi validado por usuários voluntários.

Trabalhos futuros podem investigar novas arquiteturas para aumentar a escalabilidade do sistema, como o uso de infraestrutura *serverless*. Outra direção pode ser maior investigação de características biométricas e novas *features* a serem extraídas da dinâmica de digitação. Por fim, um possível aprimoramento pode ser alcançado por aplicar modelos de aprendizado de máquina mais complexos ou mais diretamente relacionados ao tipo de dado sequencial que inicialmente é recebido como amostra.

7 REFERÊNCIAS BIBLIOGRÁFICAS

BERGADANO, F.; GUNETTI, D. ; PICARDI, C. User authentication through keystroke dynamics. **ACM Transactions on Information and System Security**, v. 5, n. 4, p. 367–397, 2002. Disponível em: <http://portal.acm.org/citation.cfm?doid=581271.581272>. Acesso em: 20 set. de 2017.

CHAVES, VICTOR VILLAS BÔAS. villasv/turbine: bare metals behind a simple yet complete and efficient Airflow setup. Disponível em: <https://github.com/villasv/turbine>. Acesso em: 20 set. de 2017.

ED DAWSON; LOPEZ, J.; MONTENEGRO, J. ; OKAMOTO, E. BAAI: biometric authentication and authorization infrastructure. In: INTERNATIONAL CONFERENCE ON INFORMATION TECHNOLOGY: RESEARCH AND EDUCATION, 2003. PROCEEDINGS. ITRE2003., 1., 2003. **Anais**... [S.l.]: IEEE, 2003, p. 274–278. Disponível em: <http://ieeexplore.ieee.org/document/1270620/>. Acesso em: 20 set. de 2017.

GUNETTI, D.; PICARDI, C. Keystroke analysis of free text. **ACM Transactions on Information and System Security**, v. 8, n. 3, p. 312–347, 2005. Disponível em: <http://portal.acm.org/citation.cfm?doid=1085126.1085129>. Acesso em: 20 set. de 2017.

HAQUE, M. A.; KHAN, N. Z. ; KHATOON, G. Authentication through keystrokes: What you type and how you type. In: PROCEEDINGS OF 2015 IEEE INTERNATIONAL CONFERENCE ON RESEARCH IN COMPUTATIONAL INTELLIGENCE AND COMMUNICATION NETWORKS, ICRCICN 2015, 1., 2016. **Anais**... [S.l.: s.n.], 2016, p. 257–261.

HASTIE, T.; TIBSHIRANI, R. ; FRIEDMAN, J. **The Elements of Statistical Learning**. New York, NY, USA: Springer New York Inc., 2001.

HU, J.; GINGRICH, D. ; SENTOSA, A. Authentication through Biometric Keystroke Dynamics. In: IEEE INTERNATIONAL CONFERENCE ON COMMUNICATIONS, 11., 2008. **Anais**... [S.l.: s.n.], 2008, p. 1556–1560. Disponível

em: <http://goanna.cs.rmit.edu.au/ jiankun/Sample_Publication/keystroke.pdf>. Acesso em: 20 set. de 2017.

KEYTRAC. KeyTrac Quickstart - KeyTrac. Disponível em: <https://www.keytrac.net/en/docs/keytrac_quickstart>. Acesso em: 20 set. de 2017.

LAU, E.; LIU, X.; XIAO, C. ; YU, X. Enhanced user authentication through keystroke biometrics. **Massachusetts Institute of Technology**, v. 9, 2004. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.529.1764 http://people.csail.mit.edu/edmond/projects/keystroke/keystroke-biometrics.pdf>. Acesso em: 20 set. de 2017.

MEIRELES, LUCAS SOUZA. LucasIME/Fingerprint: Keystroke dynamics applied to Authentication. Disponível em: <https://github.com/LucasIME/Fingerprint>. Acesso em: 20 set. de 2017.

MONROSE, F.; REITER, M. K. ; WETZEL, S. Password hardening based on keystroke dynamics. In: PROCEEDINGS OF THE 6TH ACM CONFERENCE ON COMPUTER AND COMMUNICATIONS SECURITY - CCS '99, 6., 1999. **Anais**... New York, New York, USA: ACM Press, 1999, p. 73–82.

MORALES, A.; LUNA-GARCIA, E.; FIERREZ, J. ; ORTEGA-GARCIA, J. Score normalization for keystroke dynamics biometrics. In: 2015 INTERNATIONAL CARNAHAN CONFERENCE ON SECURITY TECHNOLOGY (ICCST), 49., 2015. **Anais**... [S.l.]: IEEE, 2015, p. 223–228. Disponível em: <http://ieeexplore.ieee.org/document/7389686/>. Acesso em: 20 set. de 2017.

MORRIS, R.; THOMPSON, K. Password security: a case history. **Communications of the ACM**, v. 22, n. 11, p. 594–597, 1979. Disponível em: <http://portal.acm.org/citation.cfm?doid=359168.359172>. Acesso em: 20 set. de 2017.

MOSKOVITCH, R.; FEHER, C.; MESSERMAN, A.; KIRSCHNICK, N.; MUSTAFIC, T.; CAMTEPE, A.; LOHLEIN, B.; HEISTER, U.; MOLLER, S.; ROKACH, L. ; ELOVICI, Y. Identity theft, computers and behavioral biometrics. In: 2009 IEEE INTERNATIONAL CONFERENCE ON INTELLIGENCE AND SECURITY INFORMATICS, 7., 2009. **Anais**... [S.l.]: IEEE, 2009, p. 155–160. Disponível em: <http://ieeexplore.ieee.org/document/5137288/>. Acesso em: 20 set. de 2017.

O'GORMAN, L. Comparing passwords, tokens, and biometrics for user authentication. **Proceedings of the IEEE**, v. 91, n. 12, p. 2021–2040, 2003. Disponível em: <http://ieeexplore.ieee.org/document/1246384/>. Acesso em: 20 set. de 2017.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M. ; DUCHESNAY, E. Scikit-learn: Machine learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, 2011.

SUMAN, KUMAR NITYAN. Keystroke Dynamics | Kaggle. Disponível em: <https://www.kaggle.com/knityansuman/keystroke-dynamics>. Acesso em: 20 set. de 2017.

TYPINGDNA. Your Saas Web Application with Typing Biometrics API, Keystroke Dynamics - TypingDNA. Disponível em: <https://typingdna.com/authentication-api.html>. Acesso em: 20 set. de 2017.