Teclanômade: Uma solução de autenticação para usuários de dispositivos inteligentes baseada em Teclados Nômades

Antonio L. Maia Neto, Artur Luis Fernandes, Frederico Martins
Leandro T.C. Melo, Leonardo Cotta, Luiz Felipe Z. Saggioro, Antonio A.F. Loureiro
Leonardo B. Oliveira

[1]Universidade Federal de Minas Gerais (UFMG) – Belo Horizonte, MG, Brazil
{lemosmaia, arturluis, fredmbs, ltcmelo, leonardo.cotta,
luizfzsaggioro, loureiro,leob}@dcc.ufmg.br

Abstract. *Smart devices are becoming increasingly more relevant. This growing importance calls for tools able to provide effective authentication systems between users and their respective devices. In this paper, we claim that state-of-the-art approaches are either vulnerable to known attacks or do not fully meet usability needs. To address this problem, we came up with Teclanômade, an User-to-Device authentication scheme based on itinerant keyboards. Compared to current proposals, Teclanômade improves usability by keeping the traditional relative position of keys. Privacy provision, by its turn, stems from our keyboard's nomadic nature. Specifically, privacy is preserved by making the keyboard move to a different spot on the screen each time it is activated. Our results indicate the overhead incurred by using Teclanômade is on average 0,5 seconds.*

Resumo. *Smart devices são cada vez mais presentes em nossas vidas. A relevância crescente desses dispositivos torna a concepção de mecanismos de autenticação efetivos uma questão crucial. Neste trabalho, sustentamos que propostas "estado da arte" são vulneráveis a ataques conhecidos ou não são capazes de atender as demandas de usabilidade. Assim, para abordar este problema, propusemos o Teclanômade, um esquema de autenticação de usuários em dispositivos baseado em teclados itinerantes. Quando comparado às propostas existentes, o Teclanômade é capaz de aprimorar a usabilidade ao manter a posição relativa das teclas igual à dos teclados tradicionais. O aumento da privacidade, por sua vez, é fruto da natureza nômade do teclado. Mais precisamente, a privacidade advém do fato de que as teclas aparecem em posições distintas da tela cada vez que o teclado é acionado. Nossos resultados indicam que a sobrecarga resultante do Teclanômade– no caso, o atraso na entrada do usuário para fins de autenticação – é, em média, de apenas 0,5 segundo.*

1. Introdução

Dispositivos inteligentes (*smart devices*) estão cada vez mais presentes em nosso dia a dia. Formalmente, um *smart device* é um dispositivo eletrônico conectado a outros dispositivos semelhantes via um canal de comunicação, em geral, sem fio. A tendência é que esses dispositivos sejam cada vez mais numerosos estabelecendo, portanto, um ambiente inteligente no qual artefatos como uma simples lâmpada serão, também, "inteligentes". Juntos e atuando harmonicamente, tais dispositivos constituem a assim chamada Internet das Coisas (*Internet of Things* – IoT) [Wangham et al. 2013, Ashton 2009, Atzori et al. 2010].

 ©2015 SBC — Soc. Bras. de Computação

Paralelamente ao estabelecimento desse ambiente inteligente – e os benefícios que ele acarreta – surge também certa inquietação: como garantir a segurança e privacidade dos usuários desses dispositivos? Isso porque eles armazenam uma grande quantidade de dados (*Giga* quando não *Terabytes*). Esses dados, por sua vez, carregam consigo informações potencialmente sigilosas acerca de seus usuários. Assim, concomitantemente ao advento da IoT, é de suma importância a garantia de segurança e privacidade de seus usuários.

Uma forma de proteger dados em *smartphones*, talvez o mais popular dos *smart devices*, é através de esquemas de autenticação de usuários [Smith 2001]. Na maioria dos casos, a autenticação é engendrada por meio do compartilhamento de um segredo entre o usuário e o dispositivo [Smith 2001, Todorov 2007]. Assim, no momento da autenticação, cabe ao usuário reapresentar este segredo. Um exemplo clássico desses esquemas é o desbloqueio de tela por meio da entrada de uma senha numérica (PIN – *Personal Identification Number*) (figura 1).

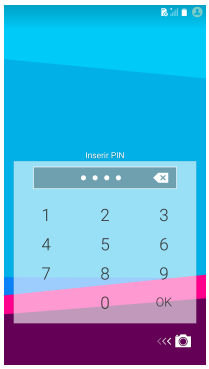


Figura 1. Desbloqueio de tela por inserção de PIN.

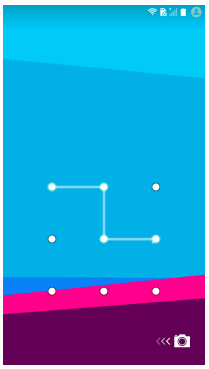


Figura 2. Desbloqueio de tela por inserção de padrão.

Tal abordagem, entretanto, não está livre de problemas. Por exemplo, uma pessoa pode observar o momento em que o código é digitado, inferi-lo e, posteriormente, usar esta informação para se passar pelo usuário legítimo [Wiedenbeck et al. 2006]. Esse problema é agravado pelo fato de que, cada vez mais, estamos cercados por câmeras; e vídeos ou imagens capturados por meio de câmeras podem ser obtidos por adversários e, em seguida, analisados cuidadosamente.

Nesta linha, o *shoulder-surfing* e o *smudge* são alguns dos ataques mais desafiadores. O ataque *shoulder-surfing* [Maggi et al. 2011] lança mão de técnicas de Visão Computacional para analisar melhor imagens gravadas por meio de câmeras amadoras (tais como aquelas presentes em *smartphones* ou *webcams*) e, assim, derivar o processo de autenticação. Este ataque: (i) estima a posição dos elementos gráficos do processo de autenticação (as teclas, por exemplo); (ii) identifica a sombra do dedo sobre a tela; (iii) aponta o local do toque e, por fim, infere o segredo. O ataque *smudge* [Aviv et al. 2010], por sua vez, utiliza técnicas de Processamento Digital de Imagem para revelar esquemas de autenticação por inserção de padrão (figura 2). Neste ataque o *software* de processamento de imagem detecta o rastro de óleo corporal deixada pelos toques do usuário na tela do dispositivo e, consequentemente, determina o padrão inserido. Os ataques *shoulder-surfing* e *smudge* chegam a atingir, respectivamente, taxas de acerto de 91.03% e 92% em

 ©2015 SBC — Soc. Bras. de Computação

certos cenários.

Objetivo. O objetivo deste trabalho é, portanto, conceber um esquema de autenticação capaz de preservar o segredo que é dado como entrada pelo usuário sem, no entanto, comprometer a facilidade de uso do dispositivo. Mais precisamente, objetivamos conceber, desenvolver e avaliar formas de inserção de senhas em teclados virtuais de *smartphones* mais resistentes a ataques.

Contribuição. Neste trabalho propusemos o Teclanômade, um esquema de autenticação inédito capaz de preservar o PIN do usuário sem, no entanto, comprometer a facilidade de uso do dispositivo. O Teclanômade é baseado em um teclado nômade, ou seja, as teclas assumem posições distintas a cada vez que o teclado é acionado. Isso, por sua vez, dificulta a ação de adversários por meio dos ataques supracitados. Quanto à usabilidade, o Teclanômade preserva a facilidade de uso ao manter a posição relativa das teclas igual a dos teclados tradicionais. Eis as principais contribuições deste artigo:

1. A concepção do Teclanômade, um teclado nômade para a entrada de PIN.
2. Uma avaliação quantitativa do esquema, em que taxas de erros e tempo de inserção do PIN foram apresentados.
3. Uma avaliação qualitativa na qual os usuários foram questionados se adotariam ou não a solução.

Os resultados indicam que o Teclanômade é capaz de aumentar o nível de segurança do esquema de autenticação sem, no entanto, degradar sua usabilidade. Em particular, o Teclanômade resultou, em média, num acréscimo de tempo de entrada de PIN de apenas 0,5 segundo. Ademais, 100% dos usuários disseram que utilizariam o Teclanômade se estivesse disponível nos dispositivos.

Organização. O restante deste trabalho é organizado da seguinte forma. Na seção 2 discutimos os trabalhos relacionados. A seção 3 apresenta o Teclanômade, isto é, nossa solução propriamente dita. Nas seções 4 e 5 são apresentados a metodologia de avaliação e os resultados, respectivamente. Por fim, concluímos o trabalho na seção 6 .

2. Trabalhos Relacionados

Há inúmeros trabalhos na área de autenticação entre usuário e aparelho. Em geral, esses trabalhos podem ser divididos em três categorias: (i) trabalhos que analisam as formas de autenticação existentes; (ii) trabalhos que propõem novos ataques contra métodos de autenticação existentes; e, por fim, (iii) trabalhos que propõem novas formas de autenticação.

O trabalho de [O'Gorman 2003] é um exemplo da primeira categoria. O autor apresenta uma comparação entre o uso de senhas, *tokens* e fatores biométricos como mecanismos de autenticação sob a perspectiva de usabilidade, segurança e custo, apresentando protocolos e ataques comuns a cada fator. O artigo [Mazurek et al. 2013] apresenta um estudo similar, analisando senhas utilizadas por mais de 25.000 docentes, funcionários e alunos universitários. Os autores mensuram a segurança de cada senha usando um algoritmo de quebra de senhas (*cracker*) e buscam uma relação entre a segurança da senha e fatores comportamentais ou demográficos dos usuários.

Um modelo de ataque a métodos de autenticação entre usuário e aparelho é o chamado *shoulder-surfing*, no qual um adversário observa o usuário utilizando seu aparelho

 ©2015 SBC — Soc. Bras. de Computação

para descobrir informações sigilosas como o conteúdo de e-mails, arquivos sendo lidos ou o segredo usado para desbloqueio do aparelho. O trabalho de [Maggi et al. 2011] apresenta um ataque de *shoulder-surfing* automático, utilizando câmeras para gravar o usuário se autenticando e um algoritmo para detectar em quais *frames* do vídeo uma tecla é pressionada e qual foi a tecla pressionada. Para o ataque, os autores consideram que o adversário não possui visão clara do teclado ou do texto digitado. O algoritmo detecta as teclas através do *popup* do caractere apresentado quando este é pressionado. Os autores comparam a precisão e velocidade do algoritmo com adversários humanos, identificando que o algoritmo é mais rápido, porém menos preciso, que ataques manuais.

O trabalho de [Andriotis et al. 2014] propõe um ataque ao mecanismo de padrão de desbloqueio (*Pattern Lock*) do Android no qual o resíduo (*smudge*) deixado na tela após a autenticação é analisado para determinar o padrão utilizado pelo usuário. Nesse ataque, o adversário tira uma foto da tela do aparelho e aplica técnicas de processamento de imagens para realçar o resíduo deixado pelo dedo do usuário. Seguindo o traço do resíduo, o adversário é capaz de determinar a sequência de pontos presentes no padrão do usuário, quebrando o sigilo da autenticação.

Similarmente, o trabalho de [Zhang et al. 2012] propõe um ataque de *smudge* para a autenticação por PIN utilizando kits de identificação de digitais para realçar os toques feitos pelo usuário. No modelo, um adversário de posse do aparelho aplica um pó específico para a identificação de digitais para realçar as marcas deixadas pelo usuário. Em seguida, o adversário fotografa a tela e utiliza técnicas de Visão Computacional para identificar quais foram os toques a partir do resíduo deixado. Por fim, esses toques são mapeados para a tela padrão de desbloqueio do telefone, para identificar quais teclas foram pressionadas.

Um novo ataque em sistemas de autenticação baseada em senhas é apresentado em [Yue et al. 2014], aplicando técnicas de Visão Computacional para detectar a senha do usuário em situações em que o adversário não consegue ver diretamente textos ou *popups* que aparecem na tela do aparelho. Neste ataque, o adversário primeiramente grava o usuário digitando sua senha e usa um algoritmo de fluxo ótico para detectar em quais quadros uma tecla é pressionada. Em seguida, bordas detectadas da tela sensível à toques são utilizadas para mapear a tela e criar uma imagem do teclado virtual. Por fim, a sombra do dedo do usuário nos quadros em que uma tecla é pressionada é usada para detectar quais pontos da tela foram tocados. Os pontos tocados são mapeados para a imagem do teclado virtual para identificar quais foram as teclas pressionadas por cada toque. Os autores obtiveram alta taxa de acerto usando câmeras de *smartphone* e *webcams*, demonstrando que o uso de equipamento especializado não é necessário para este ataque.

Ainda em [Yue et al. 2014], os autores propõem um mecanismo para prevenir o ataque proposto chamado *Privacy Enhancing Keyboard* (PEK). Duas versões do PEK são propostas. Na primeira versão, o teclado convencional é substituído por um teclado aleatório, de forma que a posição de cada caractere no teclado seja imprevisível, isso torna o adversário incapaz de gerar uma imagem do teclado virtual sem ter uma visão clara da tela do aparelho. Na segunda versão, a posição dos caracteres é atualizada repetidamente de acordo com um movimento Browniano, de forma que a posição de cada caractere é imprevisível e diferente cada vez que é pressionado. Embora essas contramedidas previnam o ataque proposto, o tempo gasto pelo usuário para se autenticar se torna também

 ©2015 SBC — Soc. Bras. de Computação

significantemente superior, sendo um prejuízo considerável para a usabilidade.

O trabalho de [Arif and Mazalek 2013] propõe uma melhoria na autenticação por senha tradicional, que consiste em uma nova forma de entrada de senha na qual os dígitos são combinados com um *stroke* (um breve movimento em uma direção). Durante a criação da senha, o usuário tem a opção de combinar cada dígito com um *stroke* na direção desejada. Com isso, o *stroke* se torna parte da senha e, no desbloqueio, a mesma combinação de dígito e *stroke* deve ser utilizada. Essa estratégia aumenta o universo de senhas numéricas de quatro dígitos para 6.250.000 possibilidades distintas. O objetivo do nosso trabalho é combinar essa estratégia com o Teclanômade para criar um método de autenticação capaz de prover maior entropia do que os métodos já existentes e que seja resistente aos ataques vistos em [Yue et al. 2014], [Andriotis et al. 2014] e [Zhang et al. 2012], enquanto tenta manter padrões visuais para aumentar a usabilidade.

Existem algumas propostas de se mitigar a taxa de sucesso de ataques *Smudge* [Andriotis et al. 2014, Zhang et al. 2012] e de Visão Computacional [Yue et al. 2014], como posicionamento aleatório dos caracteres em um teclado com senhas alfanuméricas [Yue et al. 2014], ou a combinação de toque no dígito da senha mais movimento curto (*stroke*) em alguma direção [Arif and Mazalek 2013]. Tais abordagens tem como vantagem um aumento substancial na segurança do sistema, dificultando a aplicação dos ataques nestes cenários. Em contrapartida, a usabilidade do sistema é prejudicada, já que as taxas de erro são relativamente altas e o excesso de tempo de inserção do segredo é impraticável. Nossa estratégia apresenta esse mesmo aumento de segurança, sendo resistente aos ataques citados, porém com um prejuízo menor à usabilidade, compondo um sistema seguro e com boa usabilidade.

A tabela 1 mostra uma comparação entre as estratégias de autenticação apresentadas e a nossa solução. Para a entropia, é considerado um padrão de desbloqueio de quatro números. Sobre a usabilidade, acreditamos que, mantendo os padrões visuais como proposto, nossa solução é capaz de reduzir significativamente a perda de usabilidade causada pela mudança na posição das teclas.

Método	Entropia	*Smudge*	Visão Computacional	Usabilidade
Teclado Padrão	10.000	Não	Não	-
PEK	10.000	Sim	Sim	Muito prejudicada
Teclanômade	10.000	Sim	Sim	Muito pouco prejudicada
Teclanômade com *stroke*	2.560.000	Sim	Sim	Pouco prejudicada

Tabela 1. Comparação entre métodos de autenticação.

3. Teclanômade

Nesta seção faremos uma descrição geral do Teclanômade (seção 3.1) e apresentaremos seu algoritmo com comentários sobre a implementação (seção 3.2).

3.1. Considerações iniciais

A cada ano, o número de dispositivos computacionais usados pela população aumenta. Além do considerável poder computacional, tais dispositivos carregam diversas

 ©2015 SBC — Soc. Bras. de Computação

informações do usuário, sendo muitas delas de natureza sensível.

Praticamente todos os dispositivos atualmente no mercado oferecerem recursos de segurança e/ou privacidade. No entanto, os usuários preferem não utilizar esquemas complexos que reduzem a usabilidade, mesmo que os mais simples comprometam a segurança [Raguram et al. 2011] *apud* [Arif and Mazalek 2013]. Dentre os mecanismos de autenticação tipicamente disponíveis, o mais popular é o de bloqueio de tela através de um PIN com quatro dígitos [Arif and Mazalek 2013].

É fato conhecido que esse método de autenticação é vulnerável caso ocorra perda ou roubo do PIN. O que talvez seja mais surpreendente são ataques recentemente publicados na literatura em que técnicas de Visão Computacional são empregadas para mapear os gestos realizados por usuário e, eventualmente, desvendar seu PIN [Yue et al. 2014]. Também introduzidos nos últimos anos são os ataques por detecção de *smudge*, nos quais o óleo corporal deixado pelos dedos do usuário é analisado com o fim de se rastrear as sequências de digitação mais prováveis, incluindo a do PIN de segurança [Aviv et al. 2010, Andriotis et al. 2014].

Tendo em vista a popularidade da utilização do PIN para desbloqueio de tela, seria interessante um avanço nesse mecanismo para torná-lo mais seguro sem descaracterizá-lo. A proposta PEK [Yue et al. 2014] aborda esse problema com o desenvolvimento de um teclado em que as teclas têm seus dígitos aleatoriamente distribuídos, fazendo com que o gesto realizado pelo usuário para desbloqueá-la seja frequentemente diferente, impossibilitando assim seu mapeamento. Apesar de ser mais eficiente sob a perspectiva de segurança, o PEK perde sua atratividade, pois a digitação do PIN fica mais lenta, haja vista que o usuário perde a memória visual.

Neste trabalho propomos o Teclanômade, ilustrado na figura 4, com o intuito de encontrar um melhor balanço entre segurança e usabilidade. Nosso teclado é significativamente mais robusto do que um teclado tradicional, quando analisado frente às vulnerabilidades decorrentes de ataques por Visão Computacional e por detecção de *smudge*. Isso porque suas teclas não estão posicionadas como em um teclado tradicional, mas dispersas por determinadas regiões da tela, as quais se alteram dinamicamente. Dessa forma, não é possível estabelecer um padrão preciso para o gesto realizado na digitação do PIN.

A característica chave de nosso teclado está na estratégia de dispersão das teclas na tela: ela é feita de maneira a preservar a disposição geral dos dígitos como em um teclado tradicional. Por exemplo, a tecla correspondente ao dígito 5 é sempre posicionada abaixo daquela do dígito 2, acima daquela do dígito 8, à esquerda daquela do dígito 4 e à direita daquela do dígito 6. Apesar de ter suas teclas deslocadas de suas posições originais, o Teclanômade preserva a memória visual do usuário e permite que ele digite seu PIN com agilidade.

Um suporte opcional àquele usuário que ainda assim necessita de um reforço para sua memória visual é a coloração combinada de teclas. Nesse contexto, as teclas são coloridas por linha. A figura 6 contém uma ilustração. Vale lembrar, no entanto, que a habilitação dessa funcionalidade pode enfraquecer as suposições de segurança, já que cores diferentes podem influenciar um processamento de imagem.

Na realidade, ao invés do tradicional *tap*, nosso teclado é capaz de acionar as teclas via *stroke*, conforme apresentado em [Arif and Mazalek 2013]. Esta técnica prevê

 ©2015 SBC — Soc. Bras. de Computação

que para cada entrada possível de dígito, um pequeno movimento com o dedo deve ser orientado para norte, sul, leste ou oeste. Pretendemos com isso reforçar a prevenção, não apenas a ataques de *smudge* e Visão Computacional, mas também a ataques de *shoulder-surfing* [Maggi et al. 2011].

3.2. Algoritmo

O princípio fundamental do nosso algoritmo é dispersar as teclas do Teclanômade o máximo possível, mas de maneira que a memória visual do usuário seja preservada. Em particular, buscamos manter a ordem relativa dos dígitos assim como em um teclado tradicional. Dessa forma, o dígito 1 é sempre posicionado na região mais à esquerda e mais ao alta da tela. O dígito 2, em qualquer região da tela desde que à direita do dígito 1. Já o dígito 4, em qualquer região abaixo do dígito 1. O restante dos dígitos segue o mesmo padrão de posicionamento.

O algoritmo 1 apresenta o pseudocódigo de nossa proposta. A finalidade derradeira do algoritmo que propomos é construir um *grid*, no qual é atribuído um ponto de coordenadas x e y para cada dígito, indicando a posição da respectiva tecla em um plano cartesiano, no caso, a tela do dispositivo. Sem perda de generalidade e para fins de simplicidade, assumimos um *grid* de números inteiros em um intervalo aberto $[0, \infty)$, onde a posição $(0, 0)$ corresponde ao canto superior esquerdo da tela. Uma implementação deve fazer a compactação do *grid* de acordo com a resolução desejada.

Façamos algumas definições necessárias para o entendimento do algoritmo.

Definição 1 A *linha* de uma tecla $\mathcal{T}_1$ é a *mesma* de uma tecla $\mathcal{T}_2$ se ambas, $\mathcal{T}_1$ e $\mathcal{T}_2$, são posicionadas ao longo da mesma coordenada y em um teclado tradicional. Analogamente, a *coluna* de uma tecla $\mathcal{T}_1$ é a *mesma* de uma tecla $\mathcal{T}_2$ se ambas, $\mathcal{T}_1$ e $\mathcal{T}_2$, são posicionadas ao longo da mesma coordenada x em um teclado tradicional. Por exemplo, as teclas correspondentes aos dígitos 1, 2 e 3 possuem a mesma linha, enquanto que as teclas correspondentes aos dígitos 2, 5 e 8 possuem a mesma coluna.

Definição 2 Uma *região máxima disponível*, $\mathcal{R}$, associada à tecla $\mathcal{T}$, de dígito $\mathcal{D}_T$, é a região retangular do *grid* delimitada pelo intervalo $[ax, bx]$ ao longo do eixo x e intervalo $[ay, by]$ ao longo do eixo y, tal qual os intervalos $[0, ax)$ e (bx, ∞) sejam suficientes para o posicionamento de qualquer tecla ainda não posicionada, da mesma *linha* de $\mathcal{T}$, de dígito $\mathcal{D}$; tal qual os intervalos $[0, ay)$ e (by, ∞) sejam suficientes para o posicionamento de qualquer tecla ainda não posicionada, da mesma *coluna* de $\mathcal{T}$, de dígito $\mathcal{D}$. Por exemplo, a figura 5 ilustra a região máxima disponível associada à tecla 5 quando ela é primeira tecla a ser posicionada.

Definição 3 Uma *região inválida*, $\mathcal{R}_I$, do *grid* é aquela na qual, a partir do momento de sua definição, não se pode mais posicionar nenhuma tecla.

 ©2015 SBC — Soc. Bras. de Computação

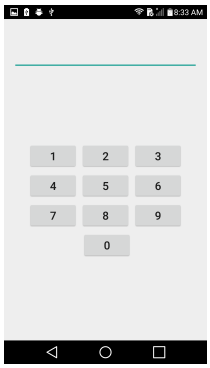


Figura 3. Teclado tradicional.

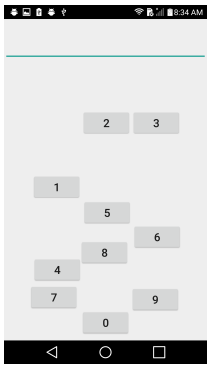


Figura 4. Teclanômade.

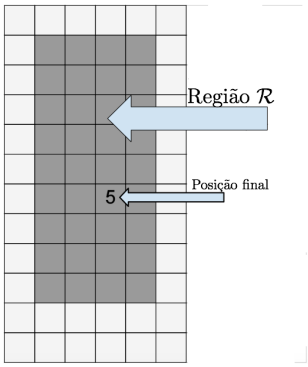


Figura 5. Exemplo de região disponível.

O algoritmo de posicionamento de teclas é parametrizado por políticas de comportamento [Alexandrescu 2001, Gamma et al. 1995]. Uma delas caracteriza qual o modelo exato para posicionamento de uma tecla dentro de sua região máxima disponível. Em nossa implementação de referência, utilizamos uma função aleatória simples.

A segunda política que parametriza nosso algoritmo é de invalidação de regiões. Quando uma região se torna inválida, o espaço reservado para as próximas regiões máximas disponíveis fica mais restrito. Portanto, esse é um fator que afeta diretamente a memória visual do usuário. Uma função com baixo limite de invalidação gera um *grid* com teclas mais dispersas, enfatizando o requisito de segurança. Já uma função com alto limite de invalidação deixa a geometria do *grid* mais próxima daquela de um teclado tradicional, enfatizando o papel da memória visual.

Nosso algoritmo recebe como entrada uma fila que determina a sequência em que as teclas são posicionadas. Naturalmente, a região máxima disponível para as teclas primeiramente posicionadas é maior do que para as teclas seguintes, resultando em teclados tendenciosos por certa preferência geométrica. Isso pode aumentar ou diminuir o conforto ao acessar determinados dígitos.

A descrição do algoritmo segue abaixo.

Algorithm 1: Posicionamento de Teclas no Grid

Entrada: Sequência $\mathcal{S}$ de teclas
Saída: Grid $\mathcal{G}$ com coordenadas para as teclas de $\mathcal{S}$

1: Construa um grid $\mathcal{G}$
2: ***Enquanto*** $\mathcal{S}$ não estiver vazia ***faça***
3: $\quad\mathcal{T} \leftarrow$ topo de $\mathcal{S}$
4: $\quad$Calcule a região disponível $\mathcal{R}$ para a tecla $\mathcal{T}$ em $\mathcal{G}$
5: $\quad\mathcal{P} \leftarrow posiciona(\mathcal{R})$
6: $\quad$Posicione $\mathcal{T}$ no ponto $\mathcal{P}$ de $\mathcal{G}$
7: $\quad\mathcal{G} \leftarrow invalida(P, R)$
8: ***Fim Enquanto***

A cada iteração, o algoritmo 1 retira uma tecla da fila, calcula sua posição e realiza a invalidação de regiões. Diretamente pelas definições, pode-se observar que sempre

 ©2015 SBC — Soc. Bras. de Computação

há uma região máxima disponível para a próxima tecla a ser processada. Uma indução simples no tamanho da fila $\mathcal{S}$ mostra que o algoritmo termina.

Para fins de prova de conceito e avaliação da usabilidade, este algoritmo foi implementado em Java para a plataforma Android.

4. Avaliação

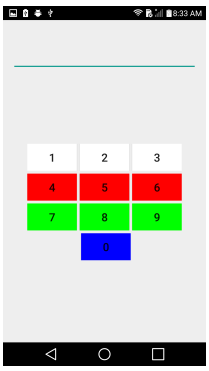
Nesta seção descrevemos a metodologia (seção 4.1) utilizada para a avaliação e condução de experimentos do Teclanômade (seção 4.2).

4.1. Metodologia

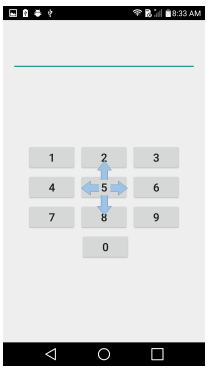
Na avaliação de nossa solução, objetivamos determinar quantitativamente a dificuldade de um usuário utilizar o Teclanômade. Para tal, imaginamos um conjunto de testes com pessoas familiarizadas com a interação com *smart devices*. Neste cenário, a forma de desbloqueio de tela mais comumente adotada é a inserção de um PIN com quatro dígitos em um teclado tradicional [Arif and Mazalek 2013]. Este é, portanto, o teclado usado como base de comparação nos experimentos. As figuras 3 e 4 apresentam um paralelo entre o teclado tradicional e o Teclanômade.

Além da comparação direta, é importante detectar o impacto das outras duas técnicas propostas neste trabalho: (i) *stroke* e (ii) coloração das linhas. Para isso, criamos versões dos teclados tradicional e Teclanômade com a aplicação individual de cada uma das técnicas e, também, com a combinação delas. A figura 6 apresenta as versões de teclado tradicional, enquanto a figura 7 apresenta as versões do Teclanômade. Como destacado nas figuras 6 e 7, nos teclados que adotam a técnica de *stroke*, há a possibilidade de movimento, $\mathcal{M}$, em quatro direções: norte ($\uparrow$), sul ($\downarrow$), leste ($\rightarrow$) e oeste ($\leftarrow$).

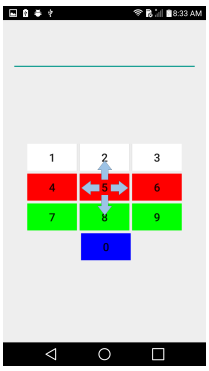
Em todos os teclados de testes, o PIN considerado é de quatro dígitos. Os dígitos do PIN são gerados aleatoriamente, assim como as quatro direções de movimento do *stroke*. Por exemplo, para os testes nos quais não há *stroke*, o PIN considerado é da forma $[\mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_3, \mathcal{D}_4]$. Já nos casos que utilizam *stroke*, o PIN tem a forma $[\mathcal{D}_1\mathcal{M}_1, \mathcal{D}_2\mathcal{M}_2, \mathcal{D}_3\mathcal{M}_3, \mathcal{D}_4\mathcal{M}_4]$, onde $\mathcal{M} \in \{\uparrow, \downarrow, \rightarrow, \leftarrow\}$.



Coloração de linhas.



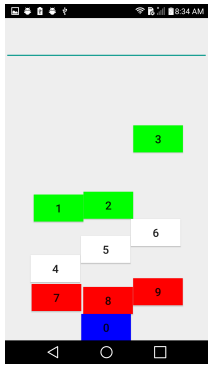
Stroke.

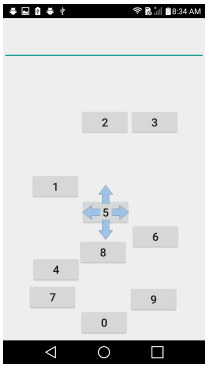


Coloração + *stroke*.

Figura 6. Diferentes variações do teclado tradicional.

 ©2015 SBC — Soc. Bras. de Computação


Coloração de linhas.


Stroke.

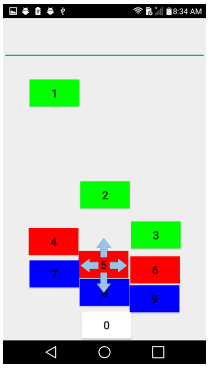

Coloração + *stroke*.

Figura 7. Diferentes variações do Teclanômade.

4.2. Experimentos

Na execução dos experimentos foram utilizados dois dispositivos: um LG G3, modelo LG-D855P, e um LG G4, modelo H815, nos quais o aplicativo de teste, contento todas as versões dos teclados, foi instalado. Antes de iniciar a inserção de PIN, os teclados foram apresentados aos voluntários e foi concedido um tempo indeterminado até que cada um deles estivesse confortável com a utilização dos diferentes teclados.

O processo de inserção de PIN seguiu os seguintes passos:

1. O PIN é gerado e mostrado em um computador externo. Durante o teste, o voluntário pode consultar o PIN quantas vezes achar necessário, isso porque não há interesse nesse trabalho em medir a capacidade de memorização do PIN;
2. O voluntário faz o cadastro do PIN no aplicativo;
3. O voluntário seleciona dentro de uma lista o teclado em que irá exercitar o PIN sorteado. A ordem dessa seleção é pré-determinada e assistida pelo instrutor do teste;
4. O usuário faz quatro entradas do PIN no teclado selecionado.

Para uma análise estatística, consideramos cinco PIN aleatórios que devem ser inseridos quatro vezes. São armazenados os tempos totais de inserção do PIN e os eventuais erros que possam ocorrer. Esse processo deve ser realizado cinco vezes para cada tipo de teclado considerado, totalizando cinco (PIN) $\times$ quatro (repetições) $\times$ oito (tipos de teclado) $=$ 160 inserções de PIN. Ao final do processo de inserções, os voluntários responderam uma pesquisa qualitativa sobre a usabilidade dos diferentes tipos de teclado.

Os experimentos foram realizados em um universo de treze pessoas destras, duas delas do sexo feminino e onze do sexo masculino. Os resultados dos experimentos são apresentados e discutidos na próxima seção.

5. Resultados

Nesta seção apresentaremos os resultados comparados com teclado tradicional (secção 5.1) e as análises de utilização do Teclanômade (seção 5.2), do *stroke* (seção 5.3) e da coloração de linhas (seção 5.4).

 ©2015 SBC — Soc. Bras. de Computação

5.1. Teclado tradicional versus Teclanômade

A figura 8 apresenta um comparativo entre variações de tempo de entrada do PIN nos teclado tradicional e Teclanômade. As taxas de erros para as variações de teclados são apresentadas na figura 9.

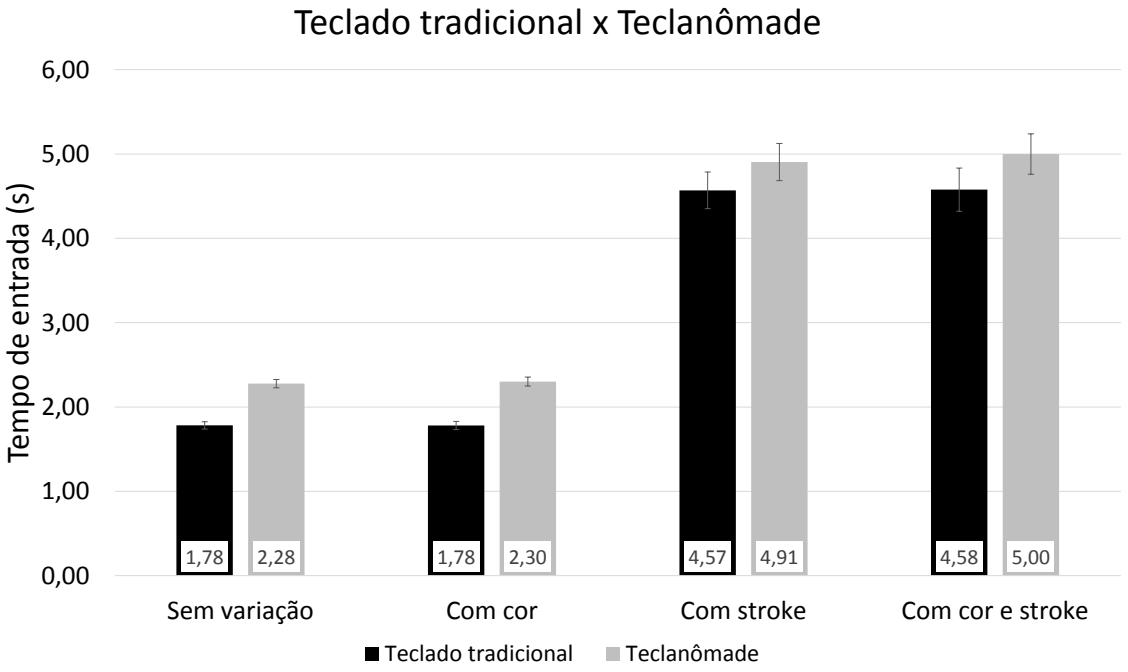


Figura 8. Comparativo do tempo de entrada do PIN no teclado tradicional e no Teclanômade.

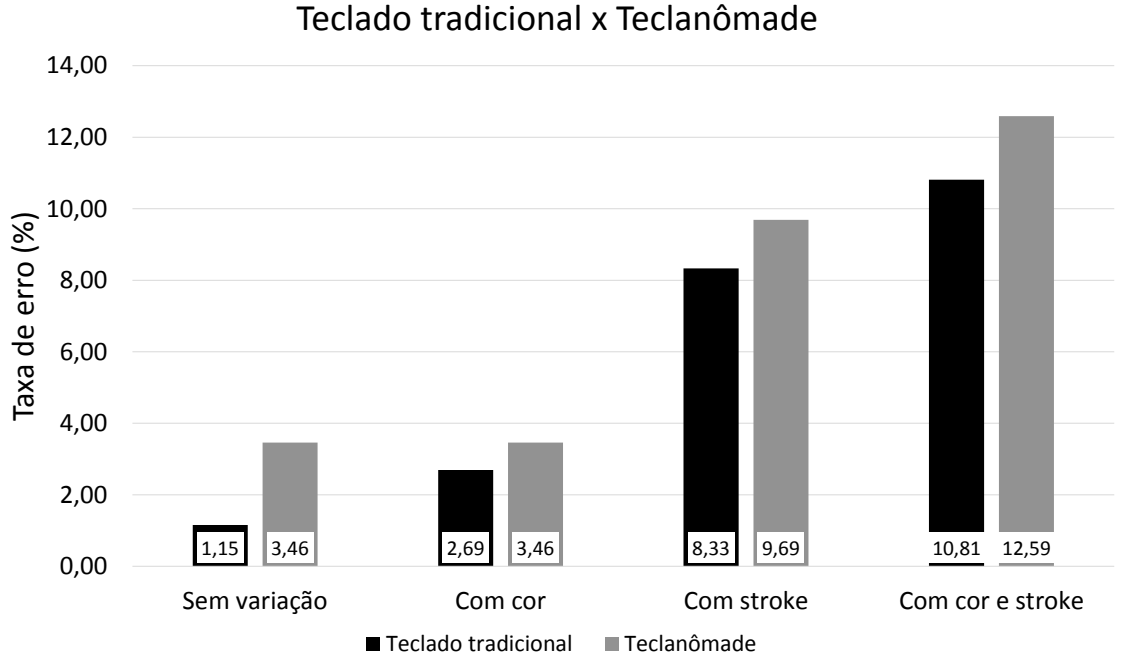


Figura 9. Comparativo da taxa de error no teclado tradicional e no Teclanômade.

©2015 SBC — Soc. Bras. de Computação

Da mesma forma como ocorre nos resultados de [Arif and Mazalek 2013], os dados apurados não seguem uma distribuição normal devido ao número insuficiente de valores distintos da amostra.

5.2. Análise do Teclanômade

Na comparação com o teclado tradicional, o uso do Teclanômade resultou em um aumento médio de apenas $0,5$ segundo no tempo de entrada. Apesar de representar um aumento proporcional de cerca de 28%, o tempo médio de atraso é perceptivelmente pequeno. Isoladamente, a taxa de erro no Teclanômade foi de apenas 3,46%.

Na avaliação qualitativa, 100% dos usuários disseram que utilizariam esse método se estivesse disponível nos dispositivos. Em termos de segurança, todos os usuários acharam o método mais seguro, sendo que dois terços (67,7%) acharam um pouco mais seguro e um terço (33,3%) acharam muito mais seguro do que o teclado tradicional.

5.3. Análise do *stroke*

Os resultados do uso do *stroke* indicaram um aumento significativo no tempo de entrada do PIN. Ocorreu um aumento de 2,5 vezes no teclado tradicional e de 2,15 vezes no Teclanômade. Apesar do aumento observado no teclado tradicional ser ligeiramente maior, o *stroke* parece afetar indistintamente ambos os teclados.

O aumento da taxa de erro com *stroke* foi ainda mais significativa: no teclado tradicional os erros aumentaram mais de 7,2 vezes, enquanto no Teclanômade os erros aumentaram 2,8 vezes. Contudo, isoladamente, as taxas de erros do uso do *stroke* foram menores que 10%. Ou seja, a taxa de acerto foi superior a 90%. Esse resultado é consistente com os obtidos em outras pesquisas, como em [Jakobsson et al. 2009].

Na pesquisa qualitativa, apenas 41,7% dos usuários disseram que usariam esse método se estivesse disponível. Ou seja, a maioria, 58,3%, disseram que não usariam o teclado com *stroke*. Por outro lado, uma ampla maioria de 83,3% dos usuários respondeu que esse método é muito mais seguro em comparação com o teclado tradicional. Somente 16,6% disseram que esse método é um pouco mais seguro. Note-se que a segurança percebida pelo método de *stroke* é maior que a do Teclanômade, contudo a intenção de uso é muito menor. Isso reforça a importância do fator usabilidade, pois mesmo que a segurança percebida e/ou efetiva seja maior, a tendência do usuário é não utilizar métodos mais complexos que reduzem a usabilidade.

Além disso, todos os usuários acharam difícil de lembrar o PIN com *stroke*. As opiniões foram divididas: 46,2% dos usuários disseram que é muito difícil lembrar o PIN, enquanto 53,8% disseram que é um pouco mais difícil lembrar o PIN com *stroke*. Nesse caso, é importante notar que o PIN com *stroke* contém oito informações, quatro números e quatro direções, enquanto os outros testes utilizaram somente quatro números. Mesmo que o usuário pudesse ver o PIN na tela do computador, a entrada do PIN com *stroke* possuía o dobro de informações. Esse fator pode ter ampliado a dificuldade de memorização relatada nos testes com *stroke*.

5.4. Análise da coloração de linhas

Os resultados indicam que não houve nenhum efeito da coloração de linhas no teclado tradicional. No Teclanômade houve um pequeno aumento do tempo de resposta do usuário.

 ©2015 SBC — Soc. Bras. de Computação

Esse resultado é consistente com os depoimentos dos usuários que também apontaram que a coloração dificultou a identificação das teclas. Provavelmente, a mudança de cor afetou de forma negativa o reconhecimento das teclas, principalmente no Teclanômade.

Na figura 9, observa-se que a adição da cor ao teclado tradicional, nas variações com e sem *stroke*, a taxa de erro aumentou em cerca de 7%. A adição da cor no Teclanômade não causou impactos na taxa de erro. A adição da cor juntamente com o *stroke* apresentou quase 3% a mais de erros em relação ao Teclanômade com *stroke*.

Esses resultados invalidam a hipótese de que a coloração poderia ajudar na usabilidade do teclado. Aliado ao fato da coloração ampliar a possibilidade de ataques visuais, conclui-se que o uso dessa técnica tende a ser desaconselhável.

6. Conclusões

A importância dos dispositivos inteligentes é cada vez maior em nosso dia a dia. Tal relevância faz com que a concepção de mecanismos de autenticação para esses dispositivos seja uma questão chave. Neste trabalho, primeiro argumentamos que as propostas "estado da arte" não foram capazes de suprir as demandas de usuários de dispositivos inteligentes. Em seguida, abordamos este problema propondo o Teclanômade, um esquema de autenticação de usuários em dispositivos baseado em teclados itinerantes. Quando comparado às propostas existentes, Teclanômade mostrou-se capaz de apresentar um bom equilíbrio entre usabilidade e privacidade. A primeira foi atingida ao se manter a posição relativa das teclas igual a dos teclados tradicionais. Já o aumento da privacidade, por sua vez, foi fruto da natureza nômade do teclado, em que as teclas aparecem em posições distintas da tela cada vez que o teclado é acionado. Nossos resultados indicaram que o atraso na entrada do usuário acarretado pelo Teclanômade é, em média, de apenas 0,5 segundo.

Referências

Alexandrescu, A. (2001). *Modern C++ Design: Generic Programming and Design Patterns Applied*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Andriotis, P., Tryfonas, T., and Yu, Z. (2014). Breaking the android pattern lock screen with neural networks and smudge attacks.

Arif, A. S. and Mazalek, A. (2013). A tap and gesture hybrid method for authenticating smartphone users. In *International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI'13)*.

Ashton, K. (2009). That 'internet of things' thing. *RFiD Journal*.

Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey. *Computer networks*.

Aviv, A. J., Gibson, K., Mossop, E., Blaze, M., and Smith, J. M. (2010). Smudge attacks on smartphone touch screens. In *4th USENIX Conference on Offensive Technologies (WOOT'10)*, pages 1–7.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-oriented Software*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

 ©2015 SBC — Soc. Bras. de Computação

Jakobsson, M., Shi, E., Golle, P., and Chow, R. (2009). Implicit authentication for mobile devices. In *Proceedings of the 4th USENIX Conference on Hot Topics in Security*, pages 9–9, Berkeley, CA, USA. USENIX Association.

Maggi, F., Volpatto, A., Gasparini, S., Boracchi, G., and Zanero, S. (2011). Poster: fast, automatic iphone shoulder surfing. In *Conference on Computer and Communications Security (CCS'11)*.

Mazurek, M., Komanduri, S., Vidas, T., Bauer, L., Christin, N., Cranor, L., Kelley, P., Shay, R., and Ur, B. (2013). Measuring password guessability for an entire university. In *Conference on Computer and Communications Security (CCS'13)*.

O'Gorman, L. (2003). Comparing passwords, tokens, and biometrics for user authentication. 91(12):2019–2040.

Raguram, R., White, A. M., Goswami, D., Monrose, F., and Frahm, J.-M. (2011). iSpy: Automatic reconstruction of typed input from compromising reflections. In *Conference on Computer and Communications Security (CCS'11)*, pages 527–536.

Smith, R. E. (2001). *Authentication: from passwords to public keys*. Addison-Wesley Longman Publishing Co., Inc.

Todorov, D. (2007). *Mechanics of user identification and authentication: Fundamentals of identity management*. CRC Press.

Wangham, M. S., Domenech, M. C., and de Mello, E. R. (2013). Infraestrutura de autenticação e de autorização para internet das coisas. In *Minicursos*, volume 1 of *13th Brazilian Symposium on Information and Computer System Security (SBSeg'13)*. SBC.

Wiedenbeck, S., Waters, J., Sobrado, L., and Birget, J.-C. (2006). Design and evaluation of a shoulder-surfing resistant graphical password scheme. In *International Working Conference on Advanced Visual Interfaces (AVI'06)*.

Yue, Q., Ling, Z., Liu, B., Fu, X., and Zhao, W. (2014). Blind recognition of touched keys on mobile devices. In *Conference on Computer and Communications Security (CCS'14)*.

Zhang, Y., Xia, P., Luo, J., Ling, Z., Liu, B., and Fu, X. (2012). Fingerprint attack against touch-enabled devices. In *2nd ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*.

 ©2015 SBC — Soc. Bras. de Computação