Segurança de aplicações baseada em informações de contexto

Bruno Assis, Eduardo Barrére

***Resumo*—A comunicação entre pessoas, dispositivos, processos e recursos, está se tornando cada vez mais frequente. A utilização de dispositivos móveis para efetuar transações bancárias, trocar informações pessoais e transferir arquivos multimídia, já é natural para os usuários. Sendo assim, se faz necessário garantir a confiabilidade e a integridade das informações trocadas através das redes de dispositivos, sejam elas públicas ou privadas. Os mecanismos de segurança atuais tendem a se tornar obsoletos a medida que os avanços tecnológicos ocorrem. Observando o crescente uso e avanço tecnológico dos dispositivos móveis, e as melhorias em seus sensores, o artigo proposto busca demonstrar a eficácia de informações contextuais, obtidas através de sensores de dispositivos, como uma ferramenta para prover um mecanismo de autenticação seguro e eficiente.**

***Palavras chave*—Segurança da informação, sensível ao contexto, dispositivos móveis.**

I. Introdução

O Rápido crescimento e evolução das diferentes tecnologias que permitem a comunicação entre pessoas, processos, máquinas ou qualquer outro dispositivo interconectado, tem facilitado o acesso à informação. Atualmente é possível acessar e-mails, ler notícias, realizar transações bancárias, enviar mensagens de texto, conversar por vídeo conferência, etc, de forma simples e de fácil acesso aos usuários, através de dispositivos móveis como *smartphones* e *tablets*. Esta mudança vem ocorrendo ao longo dos últimos anos, quando gradualmente vimos a substituição de computadores *desktops*, por dispositivos portáteis. Neste cenário, podemos observar o aumento no uso de aplicações multimídia que permitem aos usuários tirar fotos, gravar vídeos e compartilhar esses recursos entre seus dispositivos. Esta quantidade de dados carece de informações do contexto em que estão inseridos. Em uma foto, por exemplo, é interessante saber onde foi capturada, quando foi capturada e quem a capturou [1]. Assim, considerando isso podemos chamar de aplicações sensíveis ao contexto, aplicações que não só armazenam os dados multimídias, mas também utilizam informações do contexto para prover conteúdo personalizado para os usuários [2], [3]. Parte dessas aplicações podem utilizar recursos de servidores web, processamento remoto e armazenamento remoto, seja para armazenar ou trocar informações entre serviços, recursos e usuários. Assim vê-se a necessidade de garantir a integridade e a autenticidade dos dados trafegados na rede, de maneira que haja confiabilidade entre os dados recebidos e os enviados e, em caso de roubo da identidade do usuário, possam ser utilizados mecanismos de bloqueio de acesso e de informação, para que este não tenha suas informações roubadas ou sua identidade violada.

Toda aplicação, independente de plataforma que permita a troca de dados com outro recurso (através de arquiteturas Cliente-Servidor, *Peer-to-Peer* ou Híbridas), tem seus dados expostos através das redes que as interligam, se esta rede é a internet, torna-se potencialmente inevitável impedir que esses dados possam ser capturados e modificados entre o trajeto: fonte e destino. Sendo assim, um dos primeiros passos para aumentar a segurança dos dados [4], seria criptografar os dados enviados entre os participantes da rede, para que caso sejam interceptados eles não possam ser transformados em informação e utilizados por quem os capturou. Isto resolve o problema parcialmente, pois ainda deixa sem solução por exemplo, o roubo de credenciais de acesso.

Existem no mercado diversos sistemas operacionais e arquiteturas para dispositivos móveis, por exemplo: *Android*, *iOS* e *Windows Phone*. Verificamos que as linguagens de programação e as arquiteturas utilizadas nos três sistemas são diferentes em muitos aspectos. Desta forma, tratar requisitos de segurança para cada aplicação desenvolvida em cada uma destas plataformas, seria algo demorado e dispendioso.

Para encontrar uma solução viável e tentar garantir a legitimidade do usuário que se encontra autenticado na aplicação, ou que esteja enviando dados a esta, podemos utilizar recursos de aplicações sensíveis ao contexto, que nos enviam informações que podem ser utilizadas para certificar autenticidade como, por exemplo: a geolocalização do aparelho, o horário em que foram enviados os dados, informações do dispositivo que está utilizando o aplicativo, formato de mídia, entre outras. Com a posse desses dados, é possível criar heurísticas capazes de perceber se quem está trocando dados com o servidor é o dono real das credenciais.

Este artigo tem como objetivo propor uma implementação para a detecção de acessos indevidos através de heurísticas configuráveis, que utilizam informações sobre o contexto do dispositivo, assim como sugerir um algoritmo de criptografia que seja de baixo custo computacional. Estruturalmente, este artigo é apresentado da seguinte maneira: Conceitos preliminares são apresentados na Seção II, o modelo proposto é descrito na Seção III, seu funcionamento e os resultados alcançados são detalhados na Seção IV. Por fim são apresentadas as conclusões e perspectivas futuras na Seção V.

II. Fundamentação Teórica

A. Segurança da informação

A preocupação com segurança da informação não é um conceito recente e nem tampouco está restritamente rela-

Bruno Augusto Clemente de Assis, é graduado em ciência da computação pela UFJF (e-mail: brunoclemente@ice.ufjf.br)

Eduardo Barrere é Doutor em Engenharia de Sistemas e Computação pela UFRJ (e-mail: eduardo.barrere@ice.ufjf.br)

cionada à sistemas de informação. Há muito tempo, na época dos faraós já se usava conceitos de cifra de mensagens, para tentar esconder o significado de certas escrituras [5], e não tão distante podemos citar Júlio César, imperador romano que usava cifras em suas mensagens, que ficaram conhecidas como cifras de César [5], [6]. Durante a segunda guerra mundial, era de extrema necessidade que exércitos aliados pudessem se comunicar através de equipamentos de rádio, sem que a comunicação pudesse ser lida e interpretada pelos inimigos. Deste modo era necessário garantir que a informação permanecesse confidencial. Foi a partir deste momento que apareceram os primeiros avanços no ramo da criptografia. Apesar dos avanços durante a guerra, a segurança da informação só começou a se tornar destaque na sociedade civil depois da invenção das redes de computadores. Essas tecnologias permitiram abrir caminho para uma nova forma de comunicação, entretanto, trouxeram novos desafios para a segurança da informação.

B. A importância da segurança da informação

A informação vem gradualmente se tornando um quesito de primeira importância em qualquer organização. Com o advento e popularização das operações de comércio eletrônico, a informação tornou-se um ativo valioso para qualquer pessoa ou corporação, sendo imprescindível armazená-la de forma confiável [5]. Podemos dividir a segurança da informação em duas partes, a parte física, e a parte lógica [5]. Por ser muito abrangente o foco deste trabalho está diretamente relacionado à parte lógica da segurança, ou seja, tratando os requisitos somente em meios digitais.

Proteger a informação, significa que determinados pilares da segurança da informação possam ser garantidos. Alguns autores, defendem que tais pilares podem ser divididos em três grupos: confidencialidade, integridade e disponibilidade [7], ou em quatro: confidencialidade, integridade, disponibilidade e autenticidade [8], [5]. Esses pilares podem ser descritos brevemente como:

- **Confidencialidade**: A informação deve ser acessada somente por aqueles que tem prévia autorização para acessar determinado recurso.
- **Integridade**: A informação deve estar correta, ser legítima e não ter sido alterada ou danificada.
- **Disponibilidade**: A informação deve estar sempre disponível para ser acessada, caso contrário não há necessidade de se preocupar com segurança, visto que a informação já estaria inacessível e por conseguinte segura contra acessos não autorizados.
- **Autenticidade**: Devem existir mecanismos que garantam que a informação é proveniente da fonte anunciada.

Com esses pilares garantidos, pode-se começar a desenvolver sistemas mais seguros e que sejam mais confiáveis. Apesar de sua importância, a segurança deve ser balanceada com a usabilidade de um sistema de informação. Deve-se implementar níveis de segurança diferentes para cada tipo de aplicação. Por exemplo, uma aplicação de mensagem instantânea, não necessariamente precisa ter o mesmo nível de segurança de uma aplicação bancária, desta forma, mecanismos de segurança devem ser implementados, visando contemplar as diferentes necessidades de cada aplicação.

C. Criptografia

A criptografia é caracterizada por alguns autores como a arte ou ciência, na qual o objetivo principal é produzir, criar ou utilizar mecanismos que possam ocultar determinada informação, para que esta se torne ininteligível para um grande grupo de entidades, e inteligível para um pequeno e específico grupo [9], [10].

O uso da criptografia não é recente, há relatos históricos que descrevem que o processo de cifrar mensagens era usado desde a época da Grécia antiga, que ameaçada pelo reino Persa, utilizou mecanismos de cifras de mensagens para ganhar a guerra, e se livrar da iminente dominação do imperador Xerxes [4].

Durante a segunda guerra mundial, os Estados Unidos utilizaram uma técnica de criptografia simples e eficiente que resultou em grandes vitórias nas batalhas contra o Japão, a técnica consistia em fazer a comunicação entre as tropas utilizando uma linguagem muito pouco conhecida na época, a linguagem dos índios Navajo [6].

A técnica de criptografia tem como princípio transformar um texto claro (inteligível) em um texto oculto (ininteligível), utilizando cifras. Uma cifra pode ser considerada uma aplicação da criptografia, através ou não de um algoritmo [9], no qual utiliza-se uma chave com o intuito de efetuar permutações entre o conjunto de caracteres de um texto até que ele se torne ininteligível para pessoas que não estão autorizadas à acessarem seu conteúdo.

É comum pensar que os algoritmos criptográficos, não devem ser divulgados com a justificativa de que terceiros, conhecendo o algoritmo, poderiam utilizá-lo para descriptografar textos cifrados. Este é um engano comum entre os iniciantes no estudo da criptografia, pois já é difundido entre os profissionais e entusiastas de segurança da informação que, algoritmos criptográficos, devem ser sempre públicos permitindo assim que eles sejam validados, testados e verificados pelo máximo de pessoas possíveis, em busca de falhas de segurança que possam comprometê-los.

D. Criptografia por chave simétrica

A criptografia por chave simétrica é assim chamada, porque executa algoritmos criptográficos que utilizam a mesma chave, tanto para criptografar, quanto para descriptografar mensagens [6]. Tais algoritmos, utilizam o princípio de transposição ou permutação de bits para gerarem o texto criptografado. Isto significa que, quanto maior o tamanho da chave, mais difícil se torna obter o texto original. Este tipo de criptografia é frequentemente utilizada em comunicação de dados entre redes *Wi-Fi* e, ou, entre dispositivos que necessitam de segurança com baixo custo computacional [11]. Alguns algoritmos que se baseiam neste princípio são: *DES (Data Encryption Standard)* , que é atualmente considerado inseguro pois utiliza chaves de 56-bits; *AES (Advanced Encryption Standard)* , considerado o sucessor do *DES*, utiliza chaves de 128,192 e 256 bits; *3DES*

(TRIPLE DES), que se baseia em aplicar o *DES* três vezes de forma consecutiva e o *Blowfish*, que utiliza chaves entre 32 até 448 bits [12], [13].

1) Algoritmo AES: O algoritmo *AES* sigla para *Advanced Encryption Standard*, é uma cifra de bloco que permite chaves de tamanhos de 128, 192 e 256 bits e blocos com tamanhos de 128bits, é atualmente, o modelo de cifra simétrica adotado pelo governo dos Estados Unidos. Cifra de bloco consiste em dividir o texto claro em pequenos segmentos, chamados de bloco e criptografá-los separadamente, melhorando consideravelmente o desempenho do algoritmo. [10], [14]

O algoritmo *AES*, é computacionalmente seguro pois, utilizando uma chave de 128 bits, o espaço de busca seria de 2^{128} chaves, isto significa que se existisse uma máquina com 1 bilhão de processadores trabalhando em paralelo capazes de verificar uma chave a cada 1 picossegundo, seriam necessários 10^{10} anos para conseguir decifrá-la [6]. Simplificando, ninguém estaria vivo para esperar e conhecer a resposta. Além de segurança, o *AES* mostrou que possuía o melhor custo computacional, se comparado à seus concorrentes como *DES, 3DES e Blowfish* [14].

E. Criptografia por chaves assimétricas

Também chamados de algoritmos de chaves públicas, a criptografia assimétrica, trabalha de maneira diferente: o processo consiste em gerar duas chaves distintas, sendo uma chamada de chave pública, que deve ser distribuída publicamente, e uma chave privada, na qual somente uma entidade a conhece. Com a chave pública, é possível criptografar o texto claro e enviar para a entidade detentora da chave privada correspondente. A chave pública permite também, verificar a autenticidade de uma mensagem recebida pela entidade detentora da chave privada, esse processo é chamado de assinatura [6], [9]. Já a chave privada, é capaz de criptografar e descriptografar a mensagem recebida.

1) Algoritmo RSA: O *RSA* é um algoritmo de chaves assimétricas, foi apresentado pela primeira vez em 1977. O nome se dá pelas iniciais dos nomes de seus criadores - Rivest, Shamir e Adleman, todos pesquisadores do MIT na época [15]. Este algoritmo, é considerado um algoritmo muito seguro já que, sobreviveu até hoje a todas as tentativas de quebrá-lo. [6], sua eficácia é assegurada pela dificuldade computacional de efetuar fatoração e cálculo de logaritmos modulares de grandes números [9]. Suas aplicações são ilimitadas, sendo muito utilizado nos protocolos SSL/TLS para distribuição de chaves simétricas através dos navegadores, assim como o FTPs, protocolo seguro para troca de arquivos.

O processo de geração de chaves pode ser descrito em 5 passos [6]:

- Escolha de forma aleatória dois números primos grandes *P* e *Q*.
- Calcule n = $p \times q$.
- Calcule a função totiente em $\phi(n) = (p-1) \times (q-1)$.
- Escolha um inteiro *e* tal que $1 < e < \phi(n)$ de forma que *e* e $\phi(n)$ sejam primos entre si.
- Calcule *d* de forma que $d \times e \equiv 1 \times (\mod \phi(n))$.

Ao final da execução desse processo, teremos a chave pública como sendo o par de números *n* e *e*, e a chave privada o par de números *n* e *d*. Denotando *m* por um texto plano, c como sendo a mensagem cifrada e *n* e *e* como sendo a chave pública. Para criptografar uma mensagem basta calcular a função: $c = m^e \times \mod n$. Da mesma maneira, sendo (*n* e *d*) a chave privada, para descriptografar uma mensagem basta calcular a função $m = c^d \times \mod n$. A única desvantagem deste algoritmo, é que exige que chaves tenham um tamanho de pelo menos 1024 bits para ter um nível de segurança confiável, tornando-o lento em aplicações que necessitam de um rápido tempo de resposta [6].

F. Contexto

Contexto pode ser definido como sendo informações relevantes ou não, que caracterizam determinado momento em que uma aplicação ou entidade se encontram [16]. Três importantes aspectos sobre contexto podem ser enumerados como: onde você está, com quem você está e os recursos que se encontram próximo a você. Uma definição de contexto pode ser traduzida como:

> Qualquer informação que caracteriza a situação de uma entidade, sendo que, uma entidade pode ser uma pessoa, um lugar ou um objeto considerados relevantes para a interação entre um usuário e uma aplicação. O contexto é tipicamente a localização, a identidade e o estado das pessoas, grupos ou objetos físicos computacionais [18].

Segundo [17], podemos categorizar o contexto em 4 itens:

- **Contexto computacional**: rede, conectividade, custo da comunicação, banda passante, recursos (impressoras, estações, etc.).
- **Contexto do usuário**: perfil do usuário, posição, velocidade, pessoas próximas, situação social, etc.
- **Contexto físico**: luminosidade, nível de ruído, temperatura e umidade.
- **Contexto de tempo**: hora do dia, dia/mês/ano, semana, época do ano.

Tais informações e outras similares, podem ser utilizadas para criar aplicações que se comportam de maneira diferente dependendo do contexto em que estão inseridas. Essas aplicações, são comumente chamadas de aplicações sensíveis ao contexto [2].

G. Aplicações sensíveis ao contexto

Aplicações sensíveis ao contexto (ASC), são softwares que utilizam dados acerca do contexto em que estão inseridos, para tomar decisões ou fornecer informações. Em geral aplicações não sensíveis ao contexto, trabalham com os dados explicitados pelos usuários, já nas ASC, toda e qualquer informação pré-definida como contexto, é também utilizada para permitir uma maior interação entre a aplicação e o usuário, ou para fornecer dados estatísticos para o serviço fornecido. Uma ilustração deste tipo de aplicação pode ser visualizada na figura 1.

As ASC, podem utilizar diversas informações relevantes, dos mais variados dispositivos para formar um contexto, como por exemplo, os *smartphones* modernos que possuem

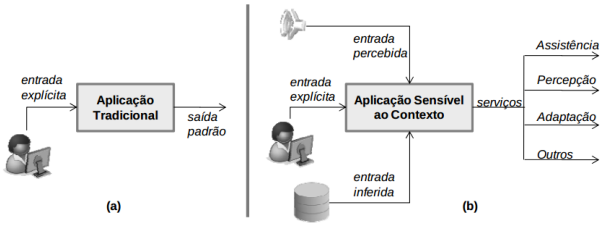

Fig. 1. Em (a) uma aplicação não sensível ao contexto, em (b) uma aplicação sensível ao contexto. [21]

diversos tipos de sensores como: de temperatura, altitude e pressão. Além disto, também carregam dados como: a geolocalização do aparelho, versão do sistema operacional, nome do dispositivo, entre outros [2]. As informações relativas ao contexto podem ajudar a desenvolver aplicações com mais interatividade e segurança para os usuários. Para exemplificar, pode-se imaginar uma aplicação para um campus universitário, ela poderia se comportar de maneira diferente em cada local da universidade, no caso do usuário estar localizado no restaurante universitário, a aplicação poderia mostrar o cardápio do dia, em contrapartida se o usuário se aproxima da biblioteca, poderia mostrar os livros com devolução em atraso.

H. Heurísticas

Heurísticas, são técnicas de solução de problemas, que tem como objetivo final obter uma solução viável, e/ou ótima a partir do prévio conhecimento de uma solução qualquer. Embora em alguns casos não seja possível encontrar a solução ótima para o problema, o uso de heurísticas pode fornecer uma solução final aproximada, tornando-a uma alternativa para a solução de problemas de alta complexidade computacional. De acordo com [19], heurísticas podem ser descritas da seguinte forma:

> Uma heurística é uma técnica que busca alcançar uma boa solução utilizando um esforço computacional considerado razoável, sendo capaz de garantir a viabilidade ou a otimalidade da solução encontrada ou, ainda, em muitos casos, ambas, especialmente nas ocasiões em que essa busca partir de uma solução viável próxima ao ótimo [19].

Em geral recomenda-se o uso de heurísticas quando, não se conhece um método de resolução exato, quando o custo computacional é alto, ou os recursos computacionais como memória e poder de processamento são escassos.

III. Modelo proposto

A proposta do módulo de segurança desenvolvido neste artigo, foi elaborada para servir também como parâmetro de implementação, para que possa ser desenvolvido em outras linguagens de programação e arquiteturas que interajam com aplicações sensíveis ao contexto.

O módulo foi desenvolvido utilizando o padrão de projetos MVC (*Model, View, Controller*) [20], este padrão permite tratar os requisitos de segurança separadamente, dividindo-os em camadas, onde o *Model* é responsável pelos objetos e consultas ao banco de dados, a *View* é responsável pela visualização do conteúdo e o *Controller*, responsável por receber requisições, enviar respostas ao cliente e também efetuar a comunicação entre o *View* e o *Model*.

É de responsabilidade do módulo de segurança, tratar todas as requisições de entrada e saída entre cliente e servidor, tratar consultas em banco de dados, efetuar autenticação dos usuários no sistema e detectar anomalias através de análise do contexto, este módulo se porta como uma nova camada no modelo MVC, efetuando um filtro de segurança entre cada camada. A figura 2, ilustra o funcionamento do modelo MVC juntamente com o módulo de segurança incorporado.

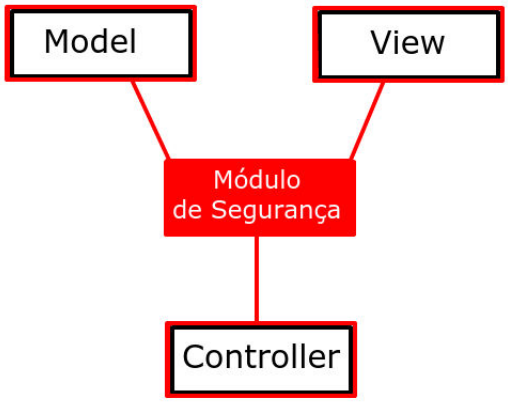

Fig. 2. Arquitetura MVC com Módulo de segurança

A. Autenticação

A autenticação através deste módulo se dá em três etapas, a primeira etapa necessita da intervenção do usuário, podendo ser atingida através de mecanismos de autenticação por login e senha, ou através de mecanismos integrados como o *OAuth2*, protocolo este que é amplamente utilizado pelo Facebook e pelo Google+. A segunda etapa, não há a intervenção do usuário, e é caracterizada pela autenticação do dispositivo no sistema, através de uma chave cadastrada, esta chave é transparente para o usuário e somente o dispositivo e o servidor a conhecem. A terceira etapa, é feita através da autenticação do contexto em que o usuário está inserido, e também não há intervenção do usuário.

B. Primeira etapa: Autenticação por Login e Senha

Esta forma de autenticação é simples e utilizada em alguns sistemas de informação, consiste em cadastrar um nome de usuário e uma senha para cada usuário em um banco de dados, e solicitar esses dois campos a cada novo acesso ao sistema, então, compara-se esses dois valores fornecidos pelo usuário com os valores armazenados no banco de dados e, caso sejam idênticos, é gerado um token de acesso ao sistema, que pode ser utilizado para identificar o usuário logado, liberando o acesso à recursos aos quais o usuário tenha permissão.

1) Autenticação por OAuth2: O OAuth2, é um protocolo de autenticação que está se tornando padrão para aplicações de redes sociais, grandes companhias como Google, Twitter e Facebook já o utilizam e fornecem APIs, para que os desenvolvedores possam desenvolver mecanismos de autenticação baseados neste protocolo.

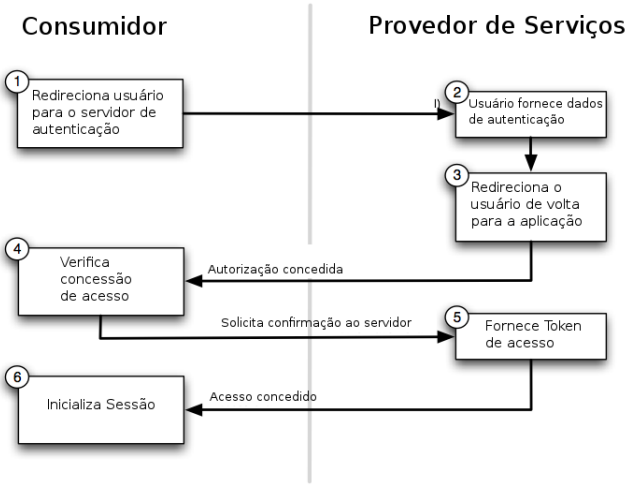

Fig. 3. Autenticação através do OAuth2 [7]

Seu funcionamento é simples, o primeiro passo do usuário é acessar uma aplicação da qual ele deseja obter recursos, após acessar a aplicação o usuário solicita acesso à um determinado recurso, como ainda não está autenticado no sistema, a aplicação redireciona o usuário para o servidor de autenticação, onde este deve fornecer seus dados de acesso. Após fornecidos e o usuário autenticado, o servidor o redireciona de volta para a aplicação, esta por sua vez, solicita uma confirmação ao servidor para verificar se foi realmente o usuário quem fez a requisição, o servidor de autenticação então responde com os dados do usuário e um token de acesso. O token recebido pode então, ser utilizado para acessar recursos da aplicação.

C. Segunda etapa: Autenticação por chave de segurança

Após feita a autenticação da primeira etapa, neste momento é solicitado uma chave de segurança, que somente o servidor de aplicação e o aplicativo instalado no dispositivo conhecem, esta etapa serve para autenticar o dispositivo, permitir o bloqueio em caso de perda ou roubo, além de negar acesso em caso da violação por terceiros, dos dados da etapa anterior. Seu funcionamento se dá da seguinte forma: ao instalar o aplicativo pela primeira vez, é gerada uma chave única que fica armazenada no dispositivo, após a execução da primeira etapa de autenticação, a chave então é enviada ao servidor que verifica sua autenticidade e, caso esta seja verdadeira, o dispositivo já foi cadastrado e liberado, sendo assim, a autenticação prossegue para a terceira etapa, que é caracterizada pela análise do contexto. Caso a chave não corresponda com a cadastrada, o sistema deve gerar uma notificação através de e-mail ou SMS para o dono da conta, perguntando se este deseja liberar este novo dispositivo, nesta notificação consta um código que será utilizado para liberar o dispositivo.

[7] $blog.rivendel.com.br/wp-content/uploads/2013/06/oauth2.png$ Acesso em: 15 nov 2014

D. Terceira etapa: Autenticação por análise de contexto

A terceira e última etapa de autenticação, é feita através da análise do contexto em que o dispositivo está inserido, e atribui uma pontuação para cada elemento pertencente ao contexto. As pontuações são concedidas baseadas nas heurísticas selecionadas, e a aprovação da autenticação é concretizada quando se obtém uma pontuação mínima, calculada com base no nível de segurança selecionado para esta etapa.

E. Análise de contexto

A análise do contexto é executada em dois momentos, no primeiro é feita a análise quando o usuário tenta se autenticar no sistema, na terceiro passo de autenticação, as outras análises de contexto, são feitas toda vez que o usuário da aplicação faz uma nova requisição ao servidor, isso garante que em caso de roubo de sessão ou falha em outros mecanismos de segurança, o contexto possa bloquear o acesso indevido a algum recurso.

A cada nova requisição, heurísticas são utilizadas para comparar o último contexto armazenado no servidor com o contexto atual enviado pelo cliente e, para cada heurística utilizada é atribuído um peso, que tem como finalidade ponderar a sua importância e ajustar a análise priorizando determinados elementos do contexto. Com base nessas heurísticas, o módulo de segurança decide se libera ou não o acesso. Se o acesso for liberado, o sistema então atualiza o contexto no banco de dados, exemplificando, imagine que um usuário localizado na cidade do Rio de Janeiro, na data de 15 de novembro às 23:05, envia um arquivo multimídia ao servidor e , logo após o envio do arquivo um outro usuário que possa ter descoberto o login, a senha e a chave do dispositivo, envia um novo arquivo um minuto depois, mas agora este usuário está localizado no Acre, às 21:05, pela diferença de localização e de fuso horário, o acesso é bloqueado, assumindo que houve uma violação de segurança.

A análise não se limita somente à geolocalização, mas também utiliza outras informações como, qual o dispositivo utilizado, qual versão do sistema operacional, velocidade de deslocamento, distância máxima da localização anterior, atraso de horário e delimitação de área. Existem inúmeras outras informações que poderiam ser utilizadas para aprimorar a segurança baseada em contexto como: pressão atmosférica, altitude, sensor de batimentos cardíacos, entre outras, porém, tais informações se restringem a dispositivos mais novos e inviabilizaria o desenvolvimento de um módulo mais genérico, como o proposto neste artigo.

F. Heurísticas

Neste trabalho, o conjunto de algoritmos que efetuam a análise do contexto, somados, constituem uma heurística, esta heurística pode ser balanceada para tentar obter a melhor solução quando se trata de análise de contexto. Todos os algoritmos quando obtêm sucesso, retornam a pontuação

parametrizada nas configurações do módulo, ou 0 em caso de insucesso. As heurísticas utilizadas no módulo são descritas a seguir:

- **Distância máxima entre o contexto atual e o contexto anterior:** Verifica a diferença entre a distância do ponto geográfico atual enviado pelo usuário, e a distância do último contexto armazenado no banco de dados, se esta diferença estiver dentro do intervalo selecionado, é retornada a pontuação atribuída a esta heurística.
- **Velocidade de deslocamento:** Verifica a velocidade de deslocamento calculando a relação distância / tempo, entre a posição atual do usuário, e a do último contexto, se esta velocidade for menor ou igual do que a estabelecida no parâmetro, retorna-se a pontuação.
- **Atraso de data:** Verifica se a data da requisição atual é anterior à da última requisição, desta forma pode-se bloquear o acesso caso haja diferença entre os relógios dos dispositivos. Caso a data seja posterior, retorna-se a pontuação parametrizada.
- **Análise do dispositivo:** Verifica se o dispositivo que fez a nova requisição, é igual ao dispositivo que fez a última requisição.
- **Análise do S.O:** Verifica se o sistema operacional é idêntico ao que fez a última requisição.
- **Delimitação de área geográfica:** Pode-se também limitar o uso do aplicativo por áreas geográficas, esta heurística estabelece dois pontos para delimitar o raio de uso da aplicação, caso a geolocalização atual esteja dentro raio selecionado, retorna-se a pontuação parametrizada. A figura 4 ilustra o exemplo de um contexto X (inválido) e um Y (válido) para esta heurística.

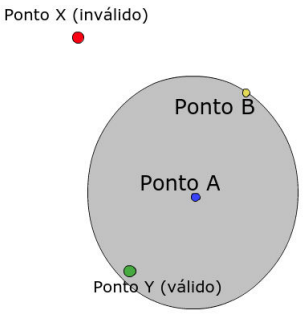

Fig. 4. Heurística por delimitação de área geográfica

Além da seleção das heurísticas, ainda pode ser parametrizado o nível de segurança a ser utilizado, possuindo três opções: 100% (Segurança máxima), 60% (média) e 30% (mínima). Supondo que cada heurística utilize 10 pontos, para se obter autenticação no sistema seriam necessários obter, para cada configuração respectivamente 60, 36 e 18 pontos, a figura 5 ilustra as possibilidades de configuração do módulo em questão.

Fig. 5. Configurações do módulo de segurança

G. Geração da chave de segurança

A chave de segurança contida no dispositivo, pode ser gerada através da combinação dos elementos do contexto no momento anterior em que é gerada, concatenando-a à um número aleatório suficientemente grande. Em seguida aplica-se o algoritmo de SHA-1 sobre esses dados, gerando uma nova chave, este processo torna a descoberta da chave por ataques por força bruta, computacionalmente difícil.

H. Segurança na comunicação de dados

Ataques de *man-in-the-middle*, que consistem em posicionar-se no meio de um trafego de dados, com o intuito de interceptar dados sigilosos, podem ser evitados utilizando criptografia para troca de informações entre cliente e servidor. Para aplicações que utilizam como cliente dispositivos móveis como, celulares e smartphones é de extrema necessidade que a criptografia, além de segura, seja eficiente, consumindo o mínimo de energia possível. Para o trabalho desenvolvido, recomenda-se o uso do protocolo SSL/TLS no servidor, juntamente com RSA utilizando uma chave de tamanho mínimo de 1024 bits para a troca das chaves simétricas, e AES com uma chave mínima de 128 bits para troca de informações entre o cliente e o servidor, pois é considerado o algoritmo simétrico mais eficiente, e de menor custo computacional [14].

IV. Validação

O estudo de caso, visa efetuar uma prova de conceito nos mecanismos de autenticação simples, por OAuth2, pela chave de segurança e pelo contexto. Atrelado a isso, são simuladas falhas na primeira, segunda e terceira etapa do processo de autenticação, detectando mudanças bruscas no contexto para novas requisições, com o objetivo de verificar a eficácia das heurísticas de análise do contexto.

A. *Simulação e configurações*

Para os testes a seguir foram utilizadas as seguintes configurações iniciais:

TABELA I
TABELA (TB_USUARIO) COM CAMPOS DO USUÁRIO INICIAL

Campo	Descrição
login	teste@moduloseguranca.com
senha	123456
Id OAuth2	800666789956826

Fig. 6. Configurações iniciais do módulo de segurança

TABELA II
TABELA (TB_CONTEXTO) COM CAMPOS DO CONTEXTO INICIAL

Campo	Descrição
Versão do S.O	7.1.2.
Dispositivo	iPhone 5
Latitude	40
Longitude	77
Data	14-11-2014 22:30:58

TABELA III
TABELA (TB_DISPOSITIVO) COM CAMPOS DO DISPOSITIVO INICIAL

Campo	Descrição
Chave	ABCD-EFGH-IJKL-MNOP
Liberado	true

B. *Simulação de perda de login e senha*

Assumindo que o usuário perdeu, ou teve seus dados da primeira etapa de autenticação violados, ou seja, perdeu seu login e senha de acesso ao sistema ou teve comprometido seus dados de acesso ao sistema de autenticação por OAuth2.

Fig. 7. Primeiro passo violado

O invasor então obtém sucesso apenas na primeira etapa, utilizando as credenciais abaixo:

Como o invasor não conhece a chave do dispositivo do usuário, o invasor efetua uma tentativa de acesso utilizando a chave arbitrária ”ABCD-ABCD-ABCD-ABCD”, o módulo de segurança bloqueia o acesso e responde com a mensagem ilustrada na figura 8.

Fig. 8. Segundo passo violado

TABELA IV
TABELA (TB_CONTEXTO) COM CAMPOS DO CONTEXTO DO ATACANTE

Campo	Descrição
Versão do S.O	7.1.2.
Dispositivo	iPhone 5
Latitude	15
Longitude	17
Data	14-11-2014 22:30:58

Como observado, o sistema identifica que a chave do dispositivo que está tentando acesso, não existe ou não está atrelada à esse usuário, e envia um e-mail ou SMS para o dono da conta informando sobre a tentativa de acesso, e também um código para liberar o dispositivo.

Após a tentativa de acesso, o sistema responde com a seguinte mensagem, representada na figura 9.

Fig. 9. Tentativa de violar o terceiro passo (contexto)

C. Simulação de perda de chave de segurança

Assumindo que além dos dados da primeira etapa, a chave do usuário também tenha sido comprometida, e que o invasor se encontra em uma região diferente da que foi feita na última requisição legítima ao servidor, sendo assim, o contexto do atacante é descrito na tabela IV, O sistema detecta que a geolocalização do usuário não contempla as configurações do módulo, bloqueia o acesso ao sistema e qualquer requisição futura.

V. Conclusão

O constante aperfeiçoamento de sensores, dispositivos móveis e equipamentos capazes de fornecer dados acerca do contexto em que estão inseridos, torna promissor o desenvolvimento de softwares que deixam a experiência do usuário mais objetiva e interativa, além de permitir a oferta de serviços e conteúdos mais precisos para cada perfil.

O desafio de desenvolver um módulo de segurança que fosse desacoplado e independente de arquitetura, foi atingido, embora limitações na quantidade de informações sobre o contexto terem estreitado o nível de segurança que poderia ser obtido, caso fosse possível utilizar mais informações como: temperatura, altitude, pressão, e etc, que são exclusivas de dispositivos mais modernos. Ficou evidenciado que, o contexto é uma ferramenta valiosa para auxiliar o desenvolvimento de medidas de seguranças. Foi possível perceber que a criptografia sozinha não soluciona todos os requisitos de segurança mas, deve ser sempre utilizada como ponto de partida, quando se inicia o desenvolvimento de qualquer projeto que exija confiabilidade.

Aliado ao uso do contexto, também fica evidente que tratar a segurança separadamente em camadas, utilizando o padrão MVC para cada etapa de autenticação (usuário, dispositivo e contexto), facilitou o levantamento de requisitos de segurança da aplicação e a implementação do projeto. Como continuidade deste deste trabalho, espera-se que o uso de contexto possa ser utilizado para prover análise de risco em transações comerciais, aliando-se à técnicas de inteligencia artificial.

Referências

[1] R. A. Feliciano, *Uma Arquitetura para um Servidor de Contexto*, Trabalho de Conclusão de Curso, Universidade Federal de Juiz de Fora, 2014.

[2] R. M. Pessoa, *Infraware: Um middleware de Suporte à aplicações sensíveis ao contexto*, Dissertação de Mestrado, Universidade Federal do Espírito Santo, 2006.

[3] Y. Feng and M. Lapata, "Automatic Image Annotation Using Auxiliary Text Information," *Proc. 46th Ann. Meeting Assoc. of Computational Linguistics: Human Language Technologies*, pp. 272-280, 2008.

[4] V. B. S. Flose, *Criptografia e Curvas Elípticas*, Dissertação de Mestrado, Universidade Estadual Paulista "Julio de Mesquita Filho", 2011.

[5] B. P. R. A. da Silva, *Planeamento e Implementação de um Sistema de Gestão da Segurança da Informação*, Dissertação de Mestrado, Universidade do Porto, 2011.

[6] A. S. Tanembaum and D. Wetheral, *Redes de Computadores*, São Paulo, Brasil, Prentice Hall, 2011, pp. 479-481.

[7] G. P. Z. Montesdioca, *Satisfação do usuário com as práticas de Segurança da Informação*, Dissertação de Mestrado, Universidade Federal do Rio Grande do Sul, 2013.

[8] J. L. P. Marciano, *Segurança da informação - uma abordagem social*, Tese de Doutorado, Universidade de Brasilia, 2006.

[9] A. Zuquete, *Segurança em redes informáticas*, São Paulo, Brasil, FCA - Editora de Informática, 2011, pp. 25-30.

[10] O. P. Verma, R. Agarwa, D. Dafouti and S. Tyagi, "Performance Analysis of Data Encryption Algorithms", presented at *Electronics Computer Technology (ICECT), 3rd International Conference*, 2011, v.5, pp. 399-403.

[11] W.Y. Zibideh and M.M. Matalgah, "Energy consumptions analysis for a class of symmetric encryption algorithm", *Radio and Wireless Symposium (RWS)*, pp. 268-270, 2014.

[12] A. A. Milad, H. Z. Muda, Z. A. B. M. Noh and M. A. Algaet, "Comparative study of performance in cryptography algorithms (Blowfish and Skipjack)", *Journal of computer sciences*, v.8, pp. 1191-1197, 2014.

[13] P. C. Mandal, "Evaluation of performance of the Symmetric Key Algorithms: DES, 3DES, AES and Blowfish", *Journal of global research in computer science*, v.3, pp. 67, 2012.

[14] M. Umaparvathi and D. K. Varughese, "Evaluation of symmetric encryption algorithms for MANETs", presented at the *Computational Intelligence and Computing Research (ICCIC), International Conference*, 2010, pp.1-3.

[15] NaQi, W. Wei, J. Zhang, W. Wang, J. Zhao, J. Li, P. Shen, X. Yin, X. Xiao and J. Hu, "Analysis and Research of the RSA Algorithm", *Information Technology Journal*. pp. 1818-1824, 2013.

[16] I. Hsu, "An architecture of mobile Web 2.0 context-aware applications in ubiquitous Web", *Journal of Software*, v.6, pp. 705-715, 2011.

[17] B. N. Schilit, N. A. and R. Want, *Context-Aware Computing Applications*, IEEE Workshop on Mobile Computing Systems and Applications, 1994.

[18] A. K. Dey and G. D. Abowd, *Towards a Better Understanding of Context and Context-Awareness*, Georgia Institute of Technology, Atlanta, GA, USA, 2000.

[19] M. C. Goldbarg and H. P. L. Luna, *Otimização combinatória e programação linear: modelos e algoritmos.* 2. Ed. Rio de Janeiro: Elsevier, 2005.

[20] E. Gamma, R. Helm, R. Johnson and J. Vlissides, *Design patterns: elements of reusable object-oriented software*, Boston, MA, USA: Addison Wesley, 1995, pp. 14.

[21] V. Vieira, P. Tedesco and A. C. Salgado. ”Modelos e Processos para o desenvolvimento de Sistemas Sensíveis ao Contexto.”, *Jornadas de Atualização em Informática*, pp. 381-431, 2009.