Design e implementação de uma arquitetura conceitual para a criação de *social botnet* em redes sociais

Vanessa Quadros G. Leite[1], **Ronaldo Moreira Salles**[1]

[1]Programa de Pós-Graduação em Sistemas e Computação – Instituto Militar de Engenharia (IME)

vanessaquadros@cos.ufrj.br, salles@ime.eb.br

Abstract. *The socialbots safety implications are evident because of the large number of users that social networks present and the high capacity of data sharing, which results in the high propagation of information. Based on this, the objective of this research is to present a conceptual architecture proposal for the creation of social botnet that can be used together with the detection tools for socialbots simulation. By implementing it on Facebook, a security analysis of this social network is performed to present the observed aspects that need to be improved by it and by the detection area of social botnet.*

Resumo. *As implicações de segurança diante dos socialbots são evidentes por causa da grande quantidade de usuários que as redes sociais apresentam e da alta capacidade de compartilhamento de dados, que traz como consequência a elevada propagação de informação. Com base nisto, o objetivo desta pesquisa é apresentar uma proposta de arquitetura conceitual para a criação de social botnet capaz de ser usada junto às ferramentas de detecção para a simulação de socialbots. Ao implementá-la no Facebook, uma análise da segurança dessa rede social é realizada para apresentar os aspectos observados que precisam ser aprimorados por ela e pela área de detecção de social botnet.*

1. Introdução

Com mais de dois bilhões de usuários ativos [Globo 2019], o Facebook[1] atrai terceiros que exploram esse meio para alcançar e coletar informações pessoais da população web [Economist 2016]. Devido a esta grande popularização, tanto ele quanto as outras *Online Social Networks* (OSNs) começaram a desempenhar novas funções [Kimurai et al. 2008, Poster 2016, DAPP 2017b, Goveia 2014].

A fim de otimizar o uso de recursos para explorar a OSN, através do uso de *socialbots*, que são softwares de automação preparados para interagir com seres humanos nas redes sociais [Ferrara et al. 2016], é possível influir pessoas em grande escala [Goga et al. 2015]. Isso é um risco principalmente em função da vulnerabilidade humana em relação aos robôs sociais exemplificada pelo compartilhamento de conteúdos falsos ou tendenciosos que muitos desses *bots* publicam [Shao et al. 2017]. E como tanto as próprias redes sociais quanto os *socialbots* podem ser usados para fins indevidos [Globo 2014, Ferrara et al. 2016], observa-se a importância de detectar e impedir que tais ações maliciosas ocorram.

[1]https://www.facebook.com/

Já no contexto de OSNs um dos problemas observados é o constante aprimoramento dos *socialbots* [Ji et al. 2016, Boshmaf et al. 2013], que faz com que os pesquisadores encontrem desafios em busca de mitigar comportamentos anômalos [Garcia-Teodoro et al. 2009], o presente artigo tem como objetivo principal propor uma arquitetura conceitual para a criação de uma *social botnet*, independente da rede social. Acredita-se que a criação e a sua implementação são meios de entender o o modo como a *social botnet* se infiltra e de salientar aspectos que devem ser considerados pelas pesquisas que atuam no escopo de detecção.

Além disso, o fato da arquitetura ser apresentada de modo conceitual e com o papel dos seus componentes bem definidos, a torna capaz de ser mantida ao longo dos anos por outros pesquisadores e permite que eles sejam capazes de simular diversos comportamentos para as ferramentas de detecção, sendo tais características os diferenciais em relação aos trabalhos relacionados abordados no capítulo 4. Já a sua implementação no Facebook funciona como uma prova de conceito de que a arquitetura é factível e também contribui na identificação de falhas na detecção de *socialbots*.

Ao discorrer sobre como a pesquisa foi realizada, este artigo apresenta a seguinte organização; o capítulo 2 define os conceitos de *socialbots* e *social botnet*; o capítulo 3 apresenta a arquitetura conceitual para criação da *social botnet* e sua implementação; o capítulo 4 descreve a execução da *social botnet* construída com base na arquitetura proposta, aborda sobre os aspectos observados durante a sua execução e também compara a arquitetura com os trabalhos relacionados; e, por fim, o capítulo 5 discorre sobre as conclusões e os trabalhos futuros.

2. Fundamentação teórica

2.1. *Socialbots*

Socialbot é um software de automação preparado para interagir com seres humanos nas redes sociais, inclusive é capaz de imitar e de influenciar o comportamentos dos usuários legítimos [Ferrara et al. 2016]. Ele pode ser categorizado com base na intenção para qual foi criado. Os respondedores automáticos utilizados para atendimento ao cliente, são exemplos de robôs sociais para fins benignos [Ferrara et al. 2016].

Já para fins malignos, consideram-se aqueles com os objetivos de enganar, explorar e manipular o discurso nas redes sociais através de diversos meios (códigos maliciosos, difamação, entre outros exemplos). O impacto dos *socialbots* não está restrito somente às OSNs, já que atinge também a sociedade [Ferrara et al. 2016, DAPP 2017a]. Um exemplo disso é a capacidade de um robô social influenciar na estabilidade do mercado financeiro [Bollen et al. 2011, DAPP 2017a].

Outra função de um *socialbot* é ser utilizado para alcançar uma posição influente e atingir mais pessoas com o seu conteúdo publicado. Isso compromete as estruturas que expressam a relação social entre perfis de uma OSN [Boshmaf et al. 2013] e representam um risco à democracia [DAPP 2017a], pois pode alterar as políticas públicas, disseminar notícias falsas e teorias conspiratórias, que geram desinformação e poluição de conteúdo [Shao et al. 2017].

2.2. *Social Botnet* (SbN)

Normalmente, o termo *social botnet* está associado a um conjunto de robôs sociais usados para fins maliciosos, devido aos casos famosos presentes na literatura. Tem-se como exemplo o Koobface [Thomas and Nicol 2010, Tanner et al. 2010], que é considerado um dos *socialbots* mais bem sucedidos, em função da sua capacidade de infectar diferentes sistemas operacionais e de capturar as credenciais dos usuários de diferentes redes sociais. É válido ressaltar que não necessariamente um *socialbot* infecta um dispositivo de usuário, já que podem ser feitos com base na criação de perfis falsos, cujo comportamento é próprio clonado de algum perfil verdadeiro [Tiwari 2017].

Uma *social botnet* (SbN) é gerenciada através do seu controlador, também denominado formalmente como um *botmaster*, representado como um indivíduo sistema mal-intencionado com acesso a uma mais contas da OSN, sendo esse o modo pelo qual interage com a rede social. As atividades que um controlador executa através de uma *social botnet* são delimitadas pelo conjunto de ações que o OSN disponibiliza para qualquer usuário verdadeiro [Compagno et al. 2015].

Entretanto, a partir do conceito de que uma SbN é um conjunto de *socialbots*, considera-se a possibilidade da mesma não estar exclusivamente associada às ações maliciosas, como nos casos em que é usada para *social marketing* [Kimurai et al. 2008]. Então, por questões éticas, a *social botnet* contruída nesta pesquisa não contempla atividades maliciosas para aquisição de contas, como disseminação de *malware*. Portanto, a abordagem utilizada neste artigo é a criação de perfis falsos.

3. Criação e implementação da arquitetura conceitual

Para criar a arquitetura conceitual capaz de construir uma *social botnet* e executar suas atividades em uma rede social, foi realizada uma divisão semântica que reflete os papéis que cada dos seus componentes desempenha. A primeira parte representa a criação de perfis, seja, contempla as atividades de criação de emails e de contas na rede social. Em função desta etapa exigir um trabalho diretamente proporcional ao tamanho da *social botnet* a ser criada, sugere-se o uso de automatizadores de testes capazes de preencher campos de página web, para que a criação de contas seja realizada de modo totalmente parcialmente automático.

É válido ressaltar que a forma como a automatização será tratada, se será parcial completa, depende da robustez da arquitetura implementada e da rede social a ser avaliada. Por exemplo, o Facebook realiza diferentes tipos de confirmação de um perfil e isso varia de acordo com o fornecedor de email. No caso do Outlook, Yahoo e Gmail, o processo é através de uma URL (*Uniform Resource Locator*) a ser acessada por email. Com outros provedores, como Prontmail, é enviado um código que deve ser digitado pelo usuário. O segundo componente refere-se à execução da *social botnet*. A partir dos componentes apresentados, foi possível obter a arquitetura conceitual exposta na Figura 1.

Nessa arquitetura não há a especificação dos componentes, somente a indicação dos papéis deles, de modo que ela possa ser aplicada em qualquer rede social que possua uma API. Isso porque na Figura 1 é sinalizada a existência de uma aplicação web que funciona sobre a respectiva API da rede social a ser explorada, com a comunicação entre

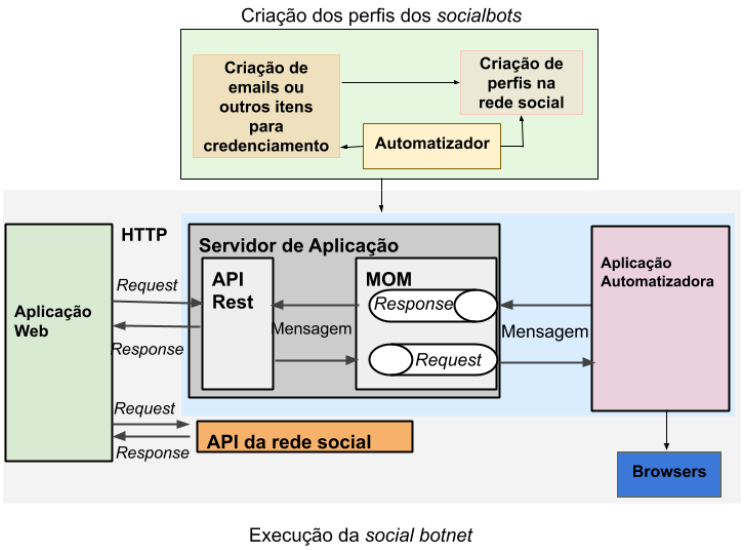

Figura 1. Estruturação das etapas "criação dos perfis dos *socialbots*"e "execução da *social botnet*"na Arquitetura Conceitual.

ambas as partes através de requisições do protocolo HTTP que utilizam uma linguagem reconhecida por *browser*, como JavaScript.

Além disso, a área delineada em azul claro permite que as demais partes do projeto evoluam independentemente umas das outras, ou seja, a ausência de acoplamento das partes do sistema garante que qualquer mudança futura impacte somente no componente a ser trabalhado. Desta forma, optou-se em propor uma arquitetura orientada a serviços.

Assim, estabeleceu-se o uso de uma API *Rest* responsável por receber as requisições HTTP da interface web, enviá-las a um *Message Oriented Middleware* (MOM), que é capaz de receber e armazenar a mensagem enviada de um lugar e de repassá-la para um outro lugar. A solicitação é enviada a um automatizador responsável por processar o pedido e respondê-lo de forma adequada. Existe também a aplicação automatizadora, que representa o código a ser desenvolvido de acordo com as ações passíveis de execução dos *socialbots* na rede social escolhida. Estas ações consistem nas funcionalidades disponibilizadas pelas redes sociais para os seus usuários.

3.1. Implementação da arquitetura conceitual

Para implementar a arquitetura, foram utilizadas algumas ferramentas apresentadas nesta seção. A fim de atribuir maior clareza, a etapa de implementação é apresentada também com base nos dois componentes semânticos: criação de perfis e *execução da social botnet*. É válido ressaltar que a implementação da arquitetura é direcionada para a exploração do Facebook por ser a rede social mais utilizada no mundo [Statista 2018]. A Figura 2 ilustra a proposta de implementação da arquitetura conceitual.

3.1.1. Implementação da etapa de criação dos perfis dos *socialbots*

Para criar uma nova conta de usuário em uma rede social, um email ou número de celular são necessários primeiro para validar e, em seguida, ativar/confirmar a conta através

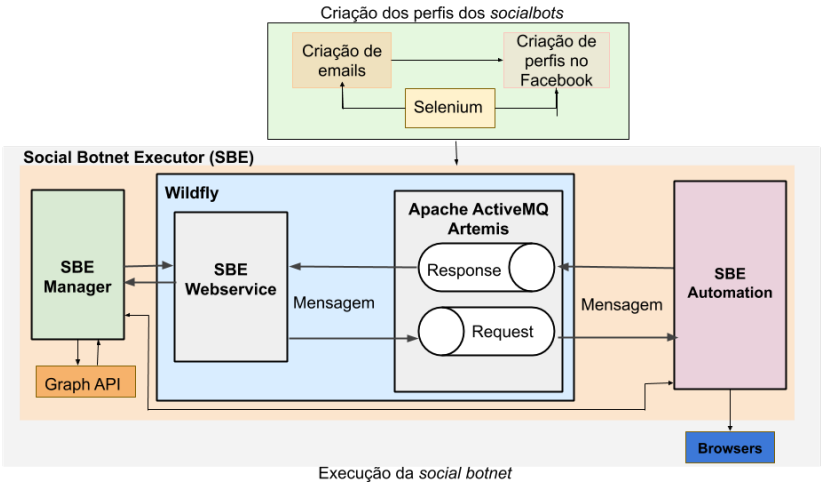

Figura 2. Proposta de implementação da arquitetura conceitual.

do email utilizado ou da inserção de um código recebido por mensagem. Consequentemente, para criar os *socialbots*, é preciso superar estes mesmos obstáculos. Em relação aos emails, existem as seguintes soluções possíveis:

- uso de emails temporários;
- utilização de provedores comuns que não limitam o número de contas de e-mail criadas por sessão de navegação ou por endereço IP (por exemplo, MailRu[2]);
- aplicação de provedores que não solicitam celulares para atestar a criação da conta, por exemplo o ProntonMail[3];
- emprego de fornecedores clássicos que solicitam celular para confirmar a criação de conta, como o Yahoo ou o Gmail;

As duas primeiras abordagens foram também descritas em [Boshmaf et al. 2013] e ainda funcionam. A terceira é efetiva e foi analisada durante a execução deste estudo. Já a última foi empregada para dificultar a detecção em função do amplo uso destes provedores. Como [Boshmaf et al. 2013] exemplifica o uso de apenas *10MinuteEmail*[4] e MailRu, para definir a solução de email temporário a ser aplicada, foram avaliadas algumas ferramentas, conforme mostra a tabela 1 .

Utilizaram-se também números temporários de celular, que não são associados a nenhum indivíduo específico e são obtidos através de sites que fornecem o serviço de recebimento de mensagem. Alguns sites, como *Receive* SMS *Online* [5] e *Receive Free* SMS *Online*[6], fornecem este tipo de serviço ao disponibilizarem um mesmo número para diversos usuários. Entretanto, quando os celulares temporários foram utilizados para criar contas, o Facebook os identificou e os classificou como inválidos ou não existentes. A única solução testada e efetiva foi o *MyTrashMobile* [7]. Essa apresenta o diferencial de disponibilizar números de telefone individuais para os seus usuários, de modo que ninguém mais os poderá utilizar.

[2]https://e.mail.ru/login
[3]https://protonmail.com/
[4]https://www.10minutemail.com/10MinuteMail/index.html?dswid=8568
[5]http://receive-sms-online.com/
[6]http://receivefreesms.com/
[7]https://pt.mytrashmobile.com

Tabela 1. Avaliação do efetividade de uso de soluções de emails no Facebook

Ferramenta	Atuação sobre o Facebook
https://temp-mail.org/	Não funciona
https://tempail.com/	Funciona
https://app.inboxbear.com/	Não funciona
https://pt.emailfake.com/	Não funciona
https://www.guerrillamail.com/	Não funciona
https://10minutemail.com/	Não funciona
https://www.mohmal.com/	Funciona
https://www.crazymailing.com/	Não Funciona

Em função de apresentar maior verificação e validação das credenciais, uma alternativa foi utilizar números celulares reais para confirmar a conta. Tais números foram adquiridos e usados mediante a autorização de seus donos, no qual também foi indicado pra eles que o seu uso era para esta parte da pesquisa. Este passo foi fundamental para facilitar o acesso dos *socialbots* ao usuários legítimos do Facebook, pois nos casos em que o número havia sido usado em uma conta verdadeira, ao associá-lo a um perfil falso, o Facebook automaticamente indicava este robô social a alguns amigos da conta original ou sugeria que o mesmo adicionasse tais amigos.

Em relação à ativação da conta, observou-se que a mesma não era impeditiva para o uso do perfil, pois o novo perfil conseguia interagir normalmente com os demais usuários e realizar as ações através das funcionalidades que o Facebook disponibiliza para os perfis legítimos. Então não houve um preocupação em automatizar este processo.

É importante ressaltar que esta primeira etapa da implementação utilizou o Selenium[8], que é um conjunto de ferramentas utilizado principalmente para automatizar testes de aplicações web. Entretanto, ele não se limita a isso, já que que pode ser usado para automatizar qualquer tipo de tarefa efetuada em um navegador [Project 2012]. Assim, através dele, foi possível automatizar o processo dpelose criação das contas de email ou dos perfis do Facebook. Nos casos onde ocorriam algum tipo de validação provedores de de emails clássicos (Gmail, Yahoo ou Outlook) ou pelo Facebook, as atividades foram executadas manualmente.

3.1.2. Implementação da etapa de execução da *social botnet*

A segunda etapa da implementação consiste na criação da ferramenta *Social Botnet Executor* (SBE), responsável por realizar as atividades da *socialbot* e fornecer uma interface de gerenciamento dos perfis falsos para o controlador da *social botnet*. Como é composta por uma estrutura mais complexa, os seus componentes de *frontend* (SBE *Manager*) e *backend* (SBE *Webservice* e SBE *Automation*) são descritos separadamente a seguir. É válido indicar que, embora todos os componentes tenham exigido uma programação referente ao seu papel na arquitetura e integração com as ferramentas utilizadas, o SBE *Automation* é o responsável por executar as ações da *social botnet* no Facebook.

[8]http://www.seleniumhq.org/

3.1.2.1. SBE *Manager*

A composição arquitetural da aplicação web foi montada de acordo com as possíveis atividades a serem realizadas por um usuário no Facebook. Desta forma, o SBE *Manager* representa a interface de gerenciamento dos *socialbots*. Através dele é possível: ativar/desativar perfis; incluir/excluir as contas dos robôs sociais na ferramenta de gerenciamento; informar qual ação será executada e o conteúdo que será aplicado. Já no Facebook, é possível fazer com que os perfis dos *socialbots* executem as atividades presentes na Tabela 2.

Tabela 2. Atividades ou ações dos perfis dos *socialbots* gerenciadas pelo SBE *Manager*.

Atividades	No próprio perfil	Em Perfis de Terceiros
Postar e curtir conteúdo	Sim	Sim
Comentar e responder comentário	Sim	Sim
Adicionar foto em um álbum	Sim	Não se aplica
Solicitar amizade ou excluir amigos	Sim	Não se aplica
Aceitar ou recusar solicitação de amizade	Sim	Não se aplica

O SBE *Manager* é composto por várias camadas, representadas pelas ferramentas que se comunicam entre si através de chamadas de funções JavaScript. Segue abaixo o detalhamento e função que cada uma das ferramentas que compõe esta interface web:

- **Servidor Apache**[9]**:** utilizado para armazenar e publicar o sistema, ou seja, ele hospeda o SBE *Manager*, de modo que ele possa ser acessado através de uma URL.
- **Pusher**[10]: foi utilizado para possibilitar envio de mensagens e comunicação assíncrona do SBE *Manager* com o SBE *Automation*, servindo como solução para notificações de recebimento de solicitações de amizade. Isso porque o SBE *Automation*, que ainda será descrito mais a frente, faz a verificação de novas solicitações de amizade.
- **SDK JavaScript Facebook:** consiste em um conjunto de funções em JavaScript que possibilita o acesso fácil ao *login* e à *Graph* API do Facebook. Foi utilizado para obter informações, como o email e a foto, do perfil do *socialbot* que for inserido no sistema SBE para ser gerenciado pela ferramenta.

3.1.2.2. SBE *Webservice*

Ele é o encarregado por intermediar a interface web (SBE *Manager*) com o SBE *Automation*. Para o *Webservice* disponibilizar uma API *Rest*, são utilizados o servidor de aplicação Wildfly[11] e o MOM interno dele conhecido como Apache ActiveMQ Artemis[12].

[9]https://httpd.apache.org/
[10]https://pusher.com/
[11]http://wildfly.org/
[12]https://activemq.apache.org/artemis/

O Apache ActiveMQ Artemis é responsável por: integrar a aplicação SBE *Webservice* com a SBE *Automation*, buscando garantir a manutenção das mensagens, independentemente da disponibilidade do SBE *Automation* ou dele mesmo; gerenciar os dois canais de comunicação, um para envio das mensagens do *Webservice* para o *Automation*, e outro de resposta, que envia mensagens do *Automation* para o *Webservice*; receber, armazenar e entregar as mensagens vindas da API *Rest* do sistema.

3.1.2.3. SBE *Automation*

Ele é o responsável por implementar, operar e automatizar nos navegodres as ações dos *socialbots* não disponíveis pela *Graph* API, sendo que cada *browser* representa um *socialbot*. Portanto, este módulo contempla a lógica desenvolvida que permite que sejam realizadas pelos *socialbots* as mesmas ações que um usuário legítimo executa no Facebook. Desta forma, qualquer nova funcionalidade de uma rede social deverá ser implementata neste item da arquitetura. Cada ação executada na aplicação SBE *Manager* efetua uma requisição para o SBE *Webservice*, que por sua vez envia o pedido para o MOM Apache ActiveMQ Artemis. Já o SBE *Automation*, através do Apache Camel, recupera estas mensagens enviadas ao MOM e fica responsável por realizar:

- o roteamento da mensagem ao método de processamento da ação correspondente de acordo com o conteúdo da mensagem (*content-based router*). Esse é identificado através de um cabeçalho definido na mensagem enviada pela API *Rest* no formato chave-valor. A chave é definida como *action* e o valor refere-se ao nome da ação que se deseja executar. O conteúdo das ações é enviado no corpo da mensagem;
- processar e automatizar no navegador;
- gerar a resposta para a API, que por sua vez notifica a interface web;

Além disso, também é de sua responsabilidade realizar a cada um minuto a verificação: de respostas de pedidos de amizade feitos pelos perfis dos *socialbots* gerenciados e da solicitação de amizades por terceiros a estes perfis. Para implementar esta gama de funcionalidades, o SBE *Automation* utiliza as ferramentas a seguir:

- **Selenium:** automatiza as tarefas no Facebook não cobertas pela API da rede social, tais como, adicionar ou remover amigos e interagir (curtir, comentar ou compartilhar) conteúdo de terceiros.
- **Quartz** [13]**:** selecionado por possibilitar o agendamento de tarefas futuras, como efetuar uma publicação com base em dados fornecidos previamente por um arquivo de entrada a ser inserido na interface SBE *Manager*. Este arquivo de entrada contém informações, como contéudo publicado e data de execução, relacionadas às ações realizadas pelos usuários do Facebook, possibilitando a clonagem destes perfis. Este arquivo é colocado pelo controlador da *social botnet* na interface web (SBE *Manager*) através de upload e é enviado ao SBE *Webservice*. Toda requisição que o *Webservice* recebe é enviada para o Apache Artemis.
- **Apache Camel**[14]**:** usado para prover de forma fácil uma integração entre o MOM e o SBE *Automation*. Ele também direciona as mensagens para suas rotas internas.

[13] http://www.quartz-scheduler.org/
[14] http://camel.apache.org/

Entende-se por rota a lógica desenvolvida para a realização de ações específicas no Facebook.

- **Pusher:** fornece uma biblioteca para o SBE *Automation* enviar mensagem para o SBE *Manager*, sinalizando que existe uma notificação de solicitação de amizade. Isso é feito pra que o controlador da *social botnet* possa ver a solicitação de amizade. Desta forma, é possivel aceitar a amizade num processo mais otimizado.

4. Execução da *social botnet*

4.1. Experimento

Com base na implementação da arquitetura proposta no capítulo 3, realizou-se uma análise da segurança do Facebook por meio da execução da *social botnet* criada. Ela ocorreu durante 8 semanas, cujo intervalo foi definido com base nos trabalhos relacionados [Boshmaf et al. 2013]. E para que ela fosse realizada, executaram-se 140 tentativas para a criação dos *socialbots*. Logo na primeira semana do experimento, 12 dos perfis não foram utilizados de imediato, o que fez com que o Facebook os bloqueasse.

Esse aspecto observado foi importante para nortear as semanas seguintes dos testes, já que notou-se que após os perfis do Facebook serem criados, confirmados e terem um período de uso recorrente igual ou superior a uma semana, dificilmente existia alguma validação posterior em relação às ações que executavam. Isso porque em uma amostra de 100 perfis falsos com fotos e que realizavam apenas o compartilhamento de conteúdo desde durante a primeira semana, o Facebook identificou apenas 8% destes perfis. Logo a taxa de infiltração neste caso foi superior à 90%.

Desta forma, conclui-se que foram obtidas 120 tentativas de sucesso na criação dos *socialbots*. Para a criação destes 120 perfis, realizaram-se as seguintes atividades: definição dos gostos pessoais e nome dos *socialbots* para posterior criação das contas de emails e de perfis; adição de perfis com gostos semelhantes e manipulação dos *socialbots* a partir do *Social Botnet Executor* para que executassem as ações também descritas na tabela 2 do capítulo 3.

4.2. Aspectos sociais e técnicos avaliados durante o experimento

Em busca de minimizar as chances de identificação dos *socialbots*, avaliaram-se com base em [Alex 2015] as características dos perfis dos usuários brasileiros nas mídias sociais, mantendo assim a distribuição por faixa etária e gênero. Além disso, com base em [Thomas and Nicol 2010], a fim de disfarçar-se como uma conta legítima, os robôs sociais mais aprimorados interagiam com os perfis com mesmo interesse.

Neste período, observou-se que os perfis mais atraentes, sobretudo o de mulheres, apresentavam mais solicitações de amizade e interações. Por exemplo, em um perfil feminino com faixa etária inferior à 30 anos, 101 pessoas foram adicionadas em 1 dia e em apenas 4 minutos, 83 pessoas foram adicionadas sem qualquer tipo de bloqueio pelo Facebook. Houve outros casos semelhantes, no qual a quantidade total de amigos ultrapassou 200, mas o melhor desfecho encontra-se neste perfil feminino. Inclusive, foi sugerido um encontro com ele por um usuário aparentemente legítimo que repassou o seu número de cellular.

Um acontecimento inesperado foi que em um perfil falso e assíduo no Facebook apenas na primeira semana logo após o seu cadastro, criado com email temporário e sem

foto, permaneceu ativo por mais de um ano. Outro exemplo de ausência de verificação ocorreu durante a inserção de imagens de repositórios públicos, já que o Facebook não os reconheceu como falsos. Identificar fotos destes repositórios é um meio de detectar os *socialbots*.

Sobre os aspectos técnicos analisados durante o experimento, também notaram-se algumas políticas de segurança que visavam limitar ações suspeitas, sejam elas relacionadas aos provedores de email ou ao Facebook. A criação do perfil ou validação no Facebook pode ser realizada a partir de um número de celular, algo que sofre uma forte verificação, pois a identificação de números celulares temporários na maioria dos casos ocorria imediatamente.

Entretanto, uma questão observada em relação a esse tipo de validação, foi que nos cenários em que um mesmo número era utilizado em diferentes perfis (até três), o Facebook sugeria os amigos associados àquelas contas de usuários, facilitando a inserção dos *socialbots* por meio de engenharia social. Atentou-se que a verificação exista principalmente quando os perfis utilizavam como credencial o email e eram criados sem qualquer interação com o Facebook. Nestes casos, a conta era excluída em menos de uma semana ou, para utilizá-la, era necessário inserir um número de celular válido.

Em relação ao uso de uma mesma máquina com um IP fixo para o processo de automatização na criação das contas do Facebook, não observou-se nenhuma restrição, já que foi permitido criar mais de vinte contas em um único dia. Não foi possível indicar se a não realização de bloqueio existiria se fosse usada uma quantidade maior de contas. Isso porque, normalmente a limitação existia por parte dos Hotmail, Gmail ou Yahoo. O primeiro restringia a criação de cinco emails por dia por números de celulares. O segundo permitia por dia a geração de duas contas com um mesmo número de celular. Já o terceiro, no primeiro dia de uso permitiu a criação de dez contas com o mesmo celular. Posteriormente, surgiu um bloqueio que durou dois dias e, em seguida, foi permitido criar cinco contas de email por dia sem que o entrave aparecesse novamente.

Desta forma, conclui-se que a parte crítica para a geração da *social botnet* é constituída das etapas que consistem na criação das contas no Facebook e a respectiva validação das mesmas. Isso foi observado principalmente porque diferentes soluções para a criação e validação das contas foram abordadas.

4.3. Comparação entre a arquitetura proposta e outras similares presentes na literatura

Com o objetivo de analisar os aspectos positivos e negativos da arquitetura implementada neste estudo, foi realizada uma comparação entre ela e as demais soluções com atuação no Facebook. Isso foi estabelecido a fim de garantir um denonimador comum para a análise comparativa. Ressalta-se que durante o estudo observou-se uma quantidade restrita de artigos que atuam diretamente nesta OSN, em função da sua API ser mais restrita que a do Twitter [Douglas 2017].

Ainda assim, o Facebook foi escolhido para ser explorado por ser a mídia social mais utiliza no mundo [Statista 2018], o que sugere a sua importância e o alto impacto que as ações maliciosas podem gerar. Já em relação a arquitetura conceitual verificou-se a existência de detalhamento na sua descrição e a sua capacidade de implementação em diversas redes sociais.

[Jin et al. 2013] executam ataques à privacidade baseados em amigos em comum em uma rede social, utilizando grafos e um *dataset* com vários atributos sintéticos dos usuários. Assim, descobriram que a partir deste ataque, um usuário malicioso é capaz de lançar ataques de privacidade que identificam amigos e vizinhos distantes de um usuário específicio. Em função do escopo do projeto basear-se em grafos e num *dataset* com dados no Facebook, não houve a discussão sobre uma arquitetura conceitual e nem a execução da *social botnet* nesta rede social. Isso implica na ausência de interação entre os robôs sociais e os usuários legítimos.

[Boshmaf et al. 2013] analisam as vulnerabilidades de uma OSN para a criação de *social botnet* através da execução dos testes por oito semanas, utilizando uma arquitetura com base na API do Facebook e alcançando uma taxa de infiltração de 80% de sucesso de contas falsas no Facebook, resultado inferior ao apresentado neste artigo estudo.

Além disso, a criação dos *socialbots* com capacidade de publicar conteúdos de usuários legítimos selecionados aleatoriamente, a fim de imitar o comportamento humano, facilita na detecção desses, visto que nem sempre os interesses ou as ações de um *socialbot* são coerentes. Inclusive porque suas publicações podem ser contraditórias. Outro problema observado é o fato dele ser exclusivamente embasado na API do Facebook, que está cada vez mais restritiva com o passar dos anos. Portanto, nota-se a necessidade de criar uma arquitetura conceitual e implementada híbrida, como foi proposto neste artigo.

Já [Compagno et al. 2015] empregam uma abordagem com uso de *malware* para construção de *bots* no Facebook e GooglePlus. Desta forma, para tornar estes robôs ativos na *botnet* Elisa, eles devem ser utilizados por uma vítima ao interagir com alguma OSN. A restrição presente é as limitações na arquitetura conceitual e implementada, que fazem com que a *botnet* Elisa não consiga ampliar a sua rede de amigos e de atuação, em função da sua propagação ser exclusivamente através de *malware*. Além disso, eles não produzem seus próprios conteúdos e nem conseguem se relacionar com usuários legítimos definidos pelo *botmaster*, também não apresentam capacidade de clonar perfis verdadeiros, diferentemente do que é realizado neste estudo.

Em [He et al. 2015] é previsto que um *rootkit* de automação de teste web (WTAR), o qual é um *malware* que usa ferramentas de teste de automação web, seja um meio para a concepção de *socialbots* maliciosos. Algo assim também é proposto no presente estudo, mas apresentado de forma híbrida, por permitir a compatibilidade com as APIs das redes sociais, possibilitando que a *social botnet* tenha dois meios de exploração. Em [He et al. 2015], ao implementar estes robôs sociais em algumas mídias sociais (Facebook, Twitter e Weibo) e validar a ameaça, eles analisaram os comportamentos dos protótipos em um ambiente de laboratório e na Internet. No Facebook, os testes baseraram-se em realizar solicitações de amizades.

Além dos *socialbots* de [He et al. 2015] serem baseados em WTAR, independente da rede social, também podem imitar comportamento humano através de um automatizador de testes, mas sem clonar diretamente um perfil, como é feito neste artigo. A pesquisa mostrou também limitação ao utilizar um *malware* que exige que, caso a máquina fosse desligada, a inicialização fosse manual e a atividade do *bot* fosse suspensa, ou seja, este comportamento é um fator de risco para a manutenção da *social botnet* e, consequentemente, da pesquisa proposta pelos autores.

É válido ressaltar que o mesmo não ocorre com a arquitetura proposta neste artigo porque a inicialização dos *socialbots* é feita pelo controlador da *social botnet*. Para que ele perca o controle dos seus robôs, é preciso que ocorra alteração indevida de credenciais ou que o Facebook bloqueie as contas, algo que é passível de acontecer com qualquer perfil usado por robôs sociais, independente da arquitetura utilizada.

5. Considerações Finais

Neste artigo foi proposta uma arquitetura conceitual parar criar uma *social botnet* independente da rede social a ser explorada. Ela difere em termos de complexidade dos trabalhos relacionados por descrever cada um dos seus componentes, de modo que seja possível a sua implementação e evolução por outros pesquisadores na área e, ao ser implementada, por proporcionar uma série de atividades realizadas pelos *socialbots* que não foram observadas nas outras *social botnets* analisadas, como a capacidade de clonar as ações de um usuário legítimo.

Esta arquitetura também se mostrou como solução híbrida, ao utilizar tanto as APIs das redes sociais como as soluções independentes delas, sendo essa uma abordagem diferente das observadas nos trabalhos relacionados, que normalmente optavam somente por uma das opções indicadas. O uso de uma solução híbrida minimiza os riscos da arquitetura tornar-se inviável em função de alguma atualização na API ou na página web da rede social.

Além disso, através da sua implementação e da ação da *social botnet* criada, foi possível analisar a segurança do Facebook pois, ao serem aplicadas diversas abordagens para as etapas de criação e validação dos perfis, foi possível observar as deficiências de segurança desta OSN e as situações em que a validação é efetiva. Também observou-se a necessidade de alguns dos serviços que são utilizados para criar os perfis serem mais restritivos. Em função dos aspectos supracitados, mesmo com diversas tentativas de validar ou de realizar bloqueio, este artigo evidenciou que a dificuldade do Facebook em mapear os *socialbots* ainda existe, mesmo isso já sendo indicado em [Kumar and Reddy 2012], um trabalho realizado em 2012.

5.1. Trabalhos Futuros

Como trabalhos futuros, propõe-se a implementação da arquitetura conceitual proposta em outras redes sociais que permitam comunicação através de API. A partir disso, é possível analisar as possíveis técnicas de infiltração de *social botnets* e avaliar o impacto gerado em outras redes sociais. Integrar esta arquitetura com alguma ferramenta de detecção é uma forma de simular diversos comportamentos de *socialbots* para que tal solução os identifique.

Um outro aspecto a ser explorado é a inclusão de inteligência artificial (IA). Ela aprimoraria as ações manuais e automáticas deste estudo. Além disso, ela introduziria novas finalidades para a ferramenta criada, como: a transformação dos *socialbots* gerados em agentes capazes de identificar os veículos de *fake news*, já que eles estão infiltrados nas OSNs. Assim, a arquitetura adaptada poderá ser usada como uma estratégia para mitigar a disseminação de desinformação online.

Além disso, ao integrar a arquitetura proposta com o *chatbot*, que é um software para realizar uma comunicação informal entre um humano e um computador

[Shawar and Atwell 2007], torna-se possível uma conversa entre os *socialbots* com os usuários legítimos em tempo real, aprimorando ainda mais o comportamento deles e tornando ainda mais complexa a sua identificação. É válido ressaltar que embora o uso de *malware* facilitasse o processo de aquisição de contas para a *social botnet* e a sua infiltração, não foi usada tal abordagem como premissa por questões éticas.

Referências

Alex, B. (2015). 2015 brazil digital future in focus. Disponível em: https://www.comscore.com/por/Insights/Apresentacoes-e-documentos/2015/2015-Brazil-Digital-Future-in-Focus. Acesso em: 18 mai. de 2018.

Bollen, J., Mao, H., and Zeng, X. (2011). Twitter mood predicts the stock market. *Journal of Computational Science*, 2(1):1 – 8.

Boshmaf, Y., Muslukhov, I., Beznosov, K., and Ripeanu, M. (2013). Design and analysis of a social botnet. *Comput. Netw.*, 57(2):556–578.

Compagno, A., Conti, M., Lain, D., Lovisotto, G., and Mancini, L. V. (2015). Boten elisa: A novel approach for botnet c&c in online social networks. In *2015 IEEE Conference on Communications and Network Security (CNS)*, pages 74–82.

DAPP, F. (2017a). Robôs, redes sociais e política: estudo da FGV/DAPP aponta interferências ilegítimas no debate público na web. Disponível em: http://dapp.fgv.br/robos-redes-sociais-e-politica-estudo-da-fgvdapp-aponta-interferencias-ilegitimas-no-debate-publico-na-web/. Acesso em: 27 set. de 2018.

DAPP, F. (2017b). Robôs, redes sociais e política no Brasil: estudo sobre interferências ilegítimas no debate público na web, riscos à democracia e processo eleitoral de 2018. Disponível em: http://dapp.fgv.br/wp-content/uploads/2017/08/Robos-redes-sociais-politica-fgv-dapp.pdf. Acesso em: 20 ago. de 2017.

Douglas, C. G. A. (2017). Extração de dados em redes sociais usando python. Disponível em: http://www.linc.ufpa.br/fabricasistemas/cursoextracao/materiais/3.%20Twitter%20e%20Facebook%20API.pdf. Acesso em: 21 set. de 2017.

Economist, T. (2016). Could facebook influence the outcome of the presidential election? Disponível em: http://www.economist.com/blogs/economist-explains/2016/05/economist-explains-15. Acesso em: 19 ago. de 2018.

Ferrara, E., Varol, O., Davis, C., Menczer, F., and Flammini, A. (2016). The rise of social bots. *Commun. ACM*, 59(7):96–104.

Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., and Vázquez, E. (2009). Anomaly-based network intrusion detection: Techniques, systems and challenges. *computers & security*, 28(1):18–28.

Globo (2014). Sequestrador diz ter planejado crime com informações de rede social. Disponível em: http://g1.globo.com/sc/santa-catarina/noticia/2014/06/sequestrador-diz-ter-planejado-crime-com-informacoes-de-rede-social.html. Acesso em: 4 jun. de 2018.

Globo (2019). Facebook completa 15 anos com 2,3 bilhões de usuários. Disponível em: https://g1.globo.com/economia/tecnologia/noticia/2019/02/04/facebook-completa-15-anos-com-23-bilhoes-de-usuarios.ghtml. Acesso em: 18 jul. de 2019.

Goga, O., Venkatadri, G., and Gummadi, K. P. (2015). The doppelgänger bot attack: Exploring identity impersonation in online social networks. Disponível em: https://people.mpi-sws.org/ gummadi/papers/impersonators_IMC2015.pdf. Acesso em: 23 jan. de 2018.

Goveia, F. (2014). Conversas citando aécio no twitter. Disponível em: http://www.labic.net/blog/internet-2/bots-contra-a-sociedade/. Acesso em: 15 mai. de 2018.

He, Y., Zhang, G., Wu, J., and QiangL (2015). Understanding a prospective approach to designing malicious social bots. *Security and Communication Networks*, 2:1–18.

Ji, Y., He, Y., Jiang, X., Cao, J., and Qiang, L. (2016). Combating the evasion mechanisms of social bots. *Computers & Security*, 58:230–249.

Jin, L., Joshi, J. B. D., and Anwar, M. (2013). Mutual-friend based attacks in social network systems. *Comput. Secur.*, 37:15–30. 8 de ago. de 2016.

Kimurai, H., Basso, L. F. C., and Martin, D. M. L. (2008). Redes sociais e o marketing de inovações. *RAM. Revista de Administração Mackenzie*, 9:157 – 181.

Kumar, N. and Reddy, R. N. (2012). *Automatic detection of fake profiles in online social networks*. PhD thesis, National Institute of Technology Rourkela, Orissa, Índia.

Poster, T. W. (2016). Obama raised half a billion online. Disponível em: http://voices.washingtonpost.com/44/2008/11/obama-raised-half-a-billion-on.html. Acesso em: 3 mai. de 2018.

Project, S. (2012). Seleniumhq. Disponível em: https://www.seleniumhq.org/docs/. Acesso em: 4 jun. de 2018.

Shao, C., Ciampaglia, G. L., Varol, O., Flammini, A., and Menczer, F. (2017). The spread of fake news by social bots. *Nature Communications*.

Shawar, B. A. and Atwell, E. (2007). Chatbots: are they really useful? In *Ldv forum*, volume 22, pages 29–49.

Statista (2018). Facebook - statistics & facts. Disponível em: https://www.statista.com/topics/751/facebook/. Acesso em: 03 jan. de 2018.

Tanner, B. K., Warner, G., Stern, H., and Olechowski, S. (2010). Koobface: The evolution of the social botnet. In *2010 eCrime Researchers Summit*, pages 1–10.

Thomas, K. and Nicol, D. M. (2010). The koobface botnet and the rise of social malware. In *2010 5th International Conference on Malicious and Unwanted Software*, pages 63–70.

Tiwari, V. (2017). Analysis and detection of fake profile over social network. In *2017 International Conference on Computing, Communication and Automation (ICCCA)*, pages 175–179.