Santos': Algoritmo para Detecção de Ataques do Tipo *Stepping-stones*

Amanda S. Santos[1], **Marcus A.R Tenorio**[1], **Adriano A. Santos**[1], **Andrey Brito**[1]

[1]Departamento de Sistemas e Computação – Universidade Federal de Campina Grande
58.429-140 – Campina Grande – PB – Brasil

amandasouza@lsd.ufcg.edu.br, mart@lsd.ufcg.edu.br

adriano@copin.ufcg.edu.br, andrey@computacao.ufcg.edu.br

Abstract. *Currently, detection of incidents related to cyber threats is considered a challenging task. The increasing number of threats in the network makes information security as a significant topic for discussion in the security community, conducting studies that propose or implement solutions for privacy and protection of users on the network. Despite the methods to carry out attacks, the stepping-stones method stands out because it is an intrusion technique that allows maintaining the anonymity of the invaders and that uses a chain of intermediate machines, which are interconnected, through remote connections. In the present research, an algorithm for detecting and classifying intrusions by stepping-stones is proposed to sort the remote connections as coming from intruders or from legitimate users. The experiments were carried out in three stages: The first one being designated in the definition of a classification profile of the connections; The second to apply the algorithm to an international database; in addition, the third for validation with actual attacks. The results indicated a statistical significance for the profile classification (*97.5% *accuracy), in the verification process as the attack detection method (*100% *accuracy) and validation in a real environment (*95% *accuracy).*

Resumo. *Atualmente, a detecção de incidentes relacionados às ameaças cibernéticas é considerada uma tarefa desafiadora e o progressivo número de ciberataques na rede acentua a importância da segurança da informação como um tema para discussões na comunidade de segurança, conduzindo estudos que proponham ou implementem soluções de privacidade e proteção aos usuários na rede. Dentre os métodos para a realização de ataques, destaca-se o* stepping-stones*, por se tratar de uma técnica de intrusão que permite manter o anonimato dos invasores e que utiliza uma cadeia de máquinas intermediárias, e interligadas entre si, por conexões remotas. É proposto, na presente pesquisa, um algoritmo de detecção e classificação de invasões por* stepping-stones*, com a finalidade de classificar as conexões remotas como provenientes de intrusos ou de legítimos usuários. Os experimentos foram realizados em três etapas, sendo a primeira designada na definição de um perfil de classificação das conexões; a segunda para aplicação do algoritmo a uma base de dados internacional; e a terceira para validação em ataques reais. Os resultados apontaram uma significância estatística na classificação de perfis (precisão de* $97,5\%$*), no processo de verificação do método de detecção dos ataques (precisão de* 100%*) e validação em um ambiente real (precisão de* 95%*).*

 © 2017 Sociedade Brasileira de Computação

1. Introdução

O avanço do poder computacional e a popularidade da Internet acentuam a importância da segurança da informação como um tema para discussões na comunidade científica. Independente do desenvolvimento das pesquisas para implementação de mecanismos de segurança cibernética, milhões de ciberataques são bem sucedidos todos os anos e, dentre esses, muitos não são reportados ou mesmo acontecem sem o conhecimento da vítima [Herd and Kriendler 2013]. Em uma ação maliciosa, os intrusos na Internet lançam os ataques às vítimas de forma indireta e sofisticada, com o propósito de diminuir as hipóteses de serem descobertos. Um método usualmente utilizado para atingir o anonimato na rede e burlar os recursos de segurança é denominado por *stepping-stones* [Kuo et al. 2010].

A intrusão por *stepping-stones* é uma técnica que mantém o anonimato do cibercriminoso, através do envio de ataques por uma cadeia de máquinas interligadas entre si por acessos remotos [Ding and Huang 2011]. Esse ataque, propicia ao intruso, a habilidade de acessar o sistema comprometido, anonimamente, para roubar informações confidenciais e vendê-las na rede, ou mesmo criptografar os dados do computador alvo e reivindicar um resgate para a concessão de acesso aos dados pela vítima.

O primeiro procedimento citado na literatura para detectar intrusões por *stepping-stones* foi desenvolvido por [Staniford-Chen and Heberlein 1995], e conforme alguns trabalhos sugerem [Daud et al. 2015, Kumar and Gupta 2016, Huang et al. 2016], as estratégias desenvolvidas na detecção de intrusos por *stepping-stones* estão em constante aprimoramento. Entretanto, a proteção do computador da vítima é restrita às limitações dos algoritmos na composição de ataques sofisticados, fazendo necessário o surgimento de técnicas elaboradas para proteger os recursos tecnológicos dos diferentes níveis de ameaças atuais [Global 2013].

Nesse cenário, o principal objetivo deste artigo é propor um algoritmo para detectar invasões por *stepping-stones* para computadores caracterizados como alvos finais durante um ciberataque em tempo de resposta. Em comparação aos principais trabalhos relacionados, como os propostos em [Ding and Huang 2011, Zhang 2014], a abordagem atual aplica um limite de erro de [Gosset 1908] para a incidência de alerta falso-positivo ao resultado calculado, dispensando o monitoramento do tráfego de todas as máquinas envolvidas na cadeia de *stepping-stones*.

Como contribuições principais desse artigo, pode-se citar: (i) a confirmação da finalidade das técnicas propostas nos trabalhos de [Ding and Huang 2011, Zhang 2014], aplicadas em tempo real do ataque; (ii) a proposta da aplicação da margem de erro de [Gosset 1908] para suprimir os alertas atípicos provenientes de alterações na rede; (iii) a inclusão do intervalo de confiança para ampliar o índice de detecção proposto nos algoritmos anteriores; e (iv) o desenvolvimento de um algoritmo e validação em ambientes reais, por três arquiteturas distintas, e dentre estas, uma técnica configurada em diferentes países.

O artigo está organizado da seguinte forma: a Seção 2 detalha os trabalhos relacionados. A Seção 3 aborda os conceitos aplicados ao método proposto no atual estudo. A Seção 4 descreve o algoritmo proposto para a construção de um modelo de detecção de intrusos. Na Seção 5 é abordado configuração do ambiente e o detalhamento do planejamento dos experimentos. A Seção 6 apresenta considerações finais e conclui o artigo.

 © 2017 Sociedade Brasileira de Computação

2. Trabalhos Relacionados

O tema *stepping-stones* foi abordado em inúmeras técnicas descritas na literatura. Entretanto, no presente artigo, como ponto principal, foram descritos apenas os trabalhos que colaboraram diretamente para o desenvolvimento do método de detecção de intrusos.

O trabalho realizado em [Zhang and Paxson 2000], desenvolveu um método para detectar intrusos em *links* na Internet, por meio dos terminais interativos. O método introduziu a ideia de *Inter-gaps* e *Intra-gaps* por intermédio da correlação do tráfego ON/OFF, gerados pelo início ou pausa da digitação do usuário. O algoritmo desenvolvido pelos autores, considera como importantes, as informações contidas nos cabeçalhos dos pacotes, tais como tempo e tamanho, e apresentou um nível de precisão relevante em seus resultados para conexões estabelecidas em uma rede local. No entanto, o algoritmo evidenciou limitações na distinção entre *stepping-stones* para *hosts* configurados a longa distância ou em diferentes países.

A proposta apresentada em [Ding and Huang 2011], sugeriu um algoritmo para estimar a distância das conexões a partir do cálculo de ida e volta dos pacotes. Os autores introduziram os conceitos de *u*RTT, com abordagem ao conceito inicial para um método de detecção baseado nas informações coletadas no computador da vítima. A distância entre as conexões eram calculadas conforme a mensuração do *gap* existente entre a máquina do intruso e o computador a ser invadido. O índice de precisão máximo para a categorização das longas conexões foi cerca de 80% na classificação de intrusos, e a presença de 15% de valores falso positivos. Entre os restrições apresentadas pelos autores, são citadas, o número limitado de longas conexões e experimentos realizados, como a ausência de métodos para detectar *scripts* automáticos durante os ataques de *stepping-stones*.

Em [Zhang 2014], foi definido uma abordagem de detecção por meio do cruzamento de pacotes durante a comunicação das conexões. A análise de modo *offline*, decorreu da coleta de dados de cada computador envolvido em uma longa cadeia de *stepping-stones*. Embora os resultados apresentem um alto índice de detecção de intrusos e uma pequena incidência de falso positivos, o algoritmo tem limitações consideráveis. A ausência de resposta ao ataque em tempo real através da coleta de informações em todos os computadores envolvidos em uma longa conexão, apresentou-se inviável, pois, de fato, não há o conhecimento legítimo da localização de todos os *hosts* comprometidos em uma invasão por *stepping-stones*.

O presente trabalho avança nas contribuições dos algoritmos propostos em [Ding and Huang 2011, Zhang 2014] ao apresentar um limite de falha na detecção de intrusos por *stepping-stones* em uma cadeia de máquinas configuradas em diferentes países.

Este limite adiciona uma limiar aceitável de erro de [Gosset 1908], para desconsiderar os dados com valores incomuns, causados por fatores externos como, por exemplo, o *jitter* intrínseco da rede. A proposta atual assume as fragilidades dos trabalhos anteriores e alerta a incidência de longas cadeias por *stepping-stones* em tempo de ataque,através de informações coletadas exclusivamente no computador da vítima.

A proposta abordada nesse trabalho, apresenta um alto índice de precisão, mesmo em ataques realizados por *scripts* ou interação humana.

© 2017 Sociedade Brasileira de Computação

3. Detectando *Stepping-Stones*

As invasões por intermédio do método de *stepping-stones* são consideradas como uma técnica popularmente utilizada para atingir o anonimato na rede. Conforme [Kuo et al. 2010], a metodologia objetiva realizar cibercrimes mediante o uso de conexões entre máquinas previamente vulneráveis e interligadas por inúmeras comunicações remotas.

O estudo realizado em [Omar et al. 2008], definiu dois termos importantes para classificar os tipos de conexões remotas na execução de um ataque por *stepping-stones*. As longas conexões ($C_1 ... C_n$) são caracterizadas pelo estabelecimento de uma conexão remota proveniente de um intruso, estendido da máquina 1 até a máquina n, por meio de inúmeras conexões em cadeia, conforme pode ser apresentado na Figura 1 pela comunicação entre os computadores $i-1$, i e $i+1$. As curtas conexões são definidas por uma conexão remota resultante de um usuário normal C_1, estabelecida entre a máquina 1 até o computador seguinte $(i-1)$.

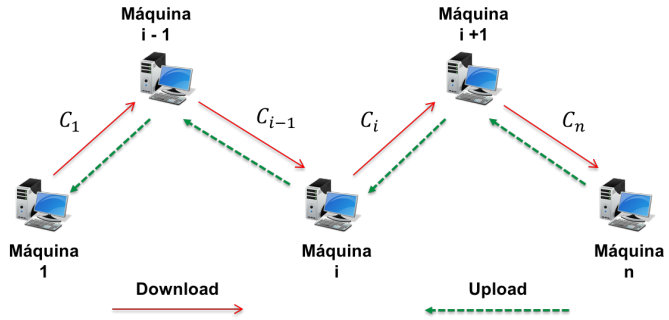


Figura 1. Conexão por *stepping-stones*

As comunicações entre os computadores em uma longa conexão, conforme apresentados na Figura 1, apresentam duas definições distintas, classificadas pelo direcionamento dos dados entre as máquinas. Para o tráfego do tipo *download*, a comunicação, por exemplo, de uma conexão por *stepping-stones*, é direcionada do *host* 1 para a máquina n. Em contrapartida, a comunicação por *upload*, é sucedida do *host* n para a máquina 1.

Para realizar uma ação maliciosa, o intruso de posse a uma coleção de contas remotas, conecta-se individualmente a computadores previamente comprometidos em outros ciberataques. Uma vez estabelecida uma longa cadeia de máquinas, entre as múltiplas seções remotas conectadas ao alvo final, conforme apresentado na Figura 2, o endereço virtual visivelmente reconhecido está diretamente ligado ao computador da vítima, dificultando o rastreamento da origem do ataque.

O exemplo contido na Figura 2, classifica as características para uma rede constituída entre as máquinas visíveis. As conexões remotas, denotam um perfil anônimo do verdadeiro endereço do cibercriminoso para a vítima, caracterizado pela visibilidade do *host* antecessor, que está conectado diretamente ao computador passível a ataques, e a omissão das demais máquinas participantes da cadeia de *stepping-stones* [Wu and Huang 2010].

 © 2017 Sociedade Brasileira de Computação

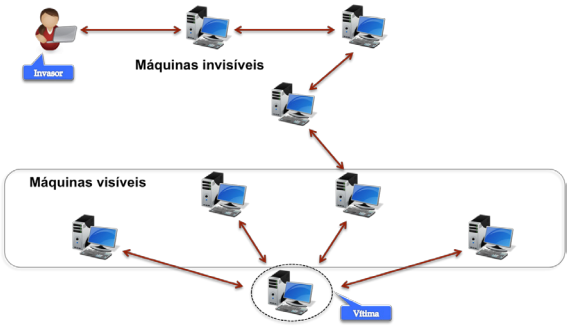


Figura 2. Proteção da vítima no final da cadeia.

3.1. Distribuição *u*RTT

O protocolo TCP possui características para a definição de mecanismos que influenciem a integridade das mensagens trocadas entre as máquinas no canal de comunicação. A nomenclatura que define os atributos na troca de mensagens em um ataque por *stepping-stones* são definidos por RTT e *u*RTT.

O atributo *Round-Trip Time* ou RTT é definido pelo tempo calculado entre a disponibilidade do envio de uma informação na rede para um destinatário, até a entrega do pacote de confirmação de recebimento ao remetente. O cálculo do RTT é estimado para definir a média de tempo que o emissor pode esperar pela confirmação de entrega da mensagem ao destinatário [Mittal et al. 2015].

A distribuição *Upload Round Trip Time* ou *u*RTT é o cálculo do tempo entre o recebimento do primeiro e o segundo pacote de requisição para o servidor. Em um ataque por *stepping-stones* estima-se um gradativo número de perda de pacotes, devido ao longo tempo para o encaminhamento da mensagem até o destinatário. Assim, o tempo da mensagem no canal de comunicação, entre as máquinas envolvidas no ataque, será maior que o tempo de ida e volta do pacote calculado pelo algoritmo do RTT [Zhang 2014].

Como um exemplo, na Figura 3 é ilustrada a troca de pacotes entre computadores envolvidos em uma cadeia de máquinas por *stepping-stones*. O computador cliente envia um pacote de requisição para a máquina-servidor. O servidor recebe o pacote requisição e envia um pacote de resposta para o cliente. O tempo calculado entre o envio do pacote requisição e o recebimento do pacote resposta é definido como RTT da conexão. O RTT pode ser apenas calculado por meio da coleta dos pacotes no computador do cibercriminoso.

Ainda na Figura 3, é definido o cálculo do tempo por *u*RTT. O cliente envia um pacote de requisição para o servidor e, este, ao receber o pacote de requisição, inicia a contagem do tempo *u*RTT e envia como confirmação, um pacote de resposta. O cliente ao receber o pacote de resposta, envia novamente um pacote de requisição para o servidor. O servidor ao considerar uma segunda requisição recebida do cliente, finaliza o cálculo do *u*RTT, determinando o valor em microssegundos. A sequência do cálculo do *u*RTT é reiniciada, e a contagem dos pacotes de requisição é zerada. O *u*RTT é calculado no computador da vítima e indispensável para detectar intrusos em um ataque por *stepping-stones*.

© 2017 Sociedade Brasileira de Computação

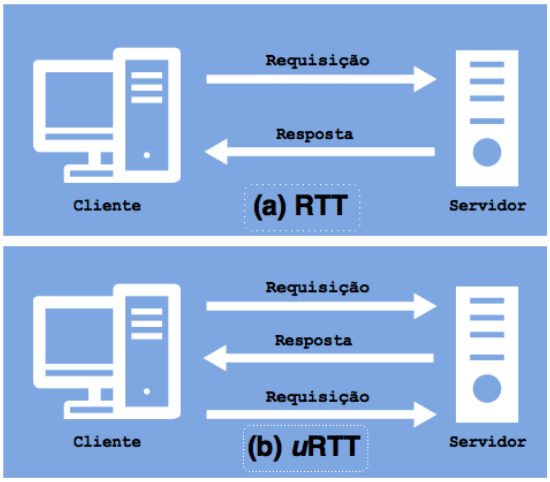


Figura 3. Distribuição RTT e *u*RTT.

3.2. Os Limites *Intra-gap* e *Inter-gap*

Para o presente estudo de detecção de intrusos por *stepping-stones*, é necessário classificar a ocorrência de ataques pelo tempo computado no *upload Round-Trip Time*, e categorizá-los em duas especificações distintas, denominadas por limites *intra-gap* e limites *inter-gap*.

A técnica proposta em [Ding and Huang 2011], permite especificar os intervalos de tempo, computados em microssegundos, para separar as informações irrelevantes dos pacotes de rede, ao qual não venham a contribuir para o algoritmo proposto.

O limite *intra-gap* está associado ao intervalo de tempo existente entre o pressionamento de teclas individuais de um comando, Unix ou Windows, e, portanto, responsáveis pela formação de um pacote de dados que será enviado como requisição para o computador da vítima.

O limite *inter-gap* consiste no espaço de tempo computado entre dois limites consecutivos. Conforme [Zhang 2014], o limite *inter-gap* consiste no intervalo de tempo entre o fim de uma sequência de caracteres determinando pela tecla *enter* e o primeiro caractere do próximo comando. Os limites *inter-gap* são importantes para estimar a existência das longas conexões, porque possui um relacionamento direto ao tamanho da cadeia de máquinas e, portanto, o *u*RTT os computadores envolvidos no ataque será progressivamente proporcional ao tamanho do número conexões definidas em uma longa cadeia de *stepping-stones*.

O exemplo apresentado na Figura 4 determina a definição de limites *intra-gap* e *inter-gap*, por meio da digitação dos comandos por um intruso. O cibercriminoso pressiona uma tecla e um pacote de requisição criptografado é criado como conteúdo para a vítima. O primeiro computador participante da cadeia *stepping-stones* recebe o pacote de requisição e, ao verificar o destinatário da mensagem, este reenvia o pacote para a próxima máquina até a informação ser recebida pela vítima. O computador da vítima, ao descriptografar o pacote recebido, infere que a informação é uma tecla inicial de um comando conhecido pelo sistema operacional, e, continua a processar os demais pacotes,

 © 2017 Sociedade Brasileira de Computação

até receber um pacote de definição para fim de comando, representado pela tecla enter. A máquina da vítima ao processar um comando, envia um pacote do tipo resposta para o computador antecessor ao ataque, e esta informação trafega o caminho inverso na cadeia de *stepping-stones*, até chegar ao computador do intruso.

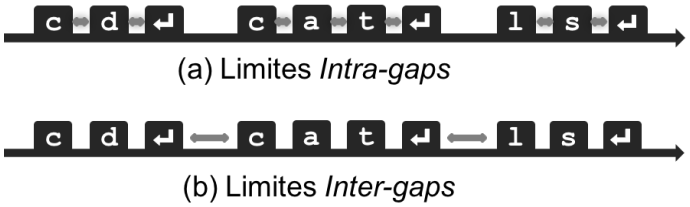


Figura 4. Os Limites *Inter-Gap* e *Intra-Gap*.

4. Algoritmo Proposto

O propósito do algoritmo é detectar a ocorrência de longas conexões a partir da observação dos dados trafegados no computador da vítima. O algoritmo, escrito na linguagem Python, adota uma ordem sequencial de passos, descritos formalmente, e combina propostas validadas em [Ding and Huang 2011, Zhang 2014], aos algoritmos matemáticos de [Gosset 1908, Pearson 1913], analisado detalhadamente abaixo.

O estágio é iniciado na captura do tráfego de rede, e para cada iteração de um experimento, o algoritmo leu os arquivos em *pcap* ou coletou os pacotes via SSH em tempo real e os classificou conforme duas categorias: pacote de requisição ou pacote de resposta. A determinação dos limites *inter-gaps* e *intra-gaps* foram calculados pelo valor de x_i, para $i = [1, ..., n]$, tal que i é o número de iterações de 1 até o n-ésimo valor da distribuição *u*RTT, ordenados em um vetor, crescentemente, pela função $sort$, tal que os valores de x_i são determinados pelo cálculo do intervalo de tempo entre primeiro t_1 e o segundo pacote de requisição t_2 na comunicação cliente e servidor, dada por

$$x_i = sort(t_2 - t_1) \tag{1}$$

onde, x_i é um vetor com valores i ordenados entre $[n, ..., 1]$.

A classificação de x_i é determinado pela exclusão dos limites *intra-gaps*, conforme as abordagens propostas em [Ding and Huang 2011, Zhang 2014]; a distinção entre os valores relacionados para o cálculo da limiar $\lambda(x)$, foram determinados pela seleção dos limites *inter-gaps* atuais x_i, combinado aos valores dos *inter-gaps* anteriores x_{i-1}, tal que $x_i \neq 0$ e $x_{i-1} > x_i$. O valor de n, determina o número atual, e é importante determinar o valor médio entre x_i e x_{i-1}, e pode ser representado por

$$\lambda(x) = \sum_{i=1}^{n} \frac{|n - x_i|}{x_{i-1}} \begin{cases} x_i \neq 0 \\ x_i < x_{i-1} \end{cases} \tag{2}$$

onde, n é número atual de valores x_i.

A distribuição mediana dos valores *inter-gaps* $D(\overline{X})$ é dado pelo cálculo da média $\overline{X}$, de um total de n valores contínuos de x_i, tal que o resultado do algoritmo possua um

© 2017 Sociedade Brasileira de Computação

valor influente acerca dos resultados discrepantes dos pacotes coletados. A equação 3 determina a distribuição da média *inter-gaps* $D(\overline{X})$, respectivamente,

$$D(\overline{X}) = \left(1 + \frac{x_{\overline{1}}^{n} + x_{\overline{2}}^{n}}{2}\right) \times \lambda(x), \tag{3}$$

sendo, x o valor atual de x_i e $\lambda(x)$ a limiar atual de x_i.

Os valores de confiança $XC_n(\lambda(x); D(\overline{X}))$ dos limites *inter-gaps* são resultados com valores padrões e aproximados ao valor exato na detecção de intrusos, e é dado por

$$XC_n(\lambda(x); D(\overline{X})) = \overline{X} \left(\sum_{i=1}^{n} \frac{|x_i \times D(\overline{X}) - x_{i-1}|}{n}\right) \begin{cases} x_i \neq 0 \\ x_i < x_{i-1} \end{cases} \tag{4}$$

O intervalo de confiança IC_f estabelece uma estimativa de valores prováveis na detecção de intrusos por *stepping-stone*, para um nível de confiança de 99% na distribuição *T-test* de [Gosset 1908], determinado por $\frac{\sigma}{\sqrt{n}}$ para erro da média $\mu(XC_n)$, dado por,

$$IC_f = \left(\mu(XC_n) - 2,5 * \frac{\sigma}{\sqrt{n}},\ \mu(XC_n) + 2,5 * \frac{\sigma}{\sqrt{n}}\right), \tag{5}$$

tal que, μ representa a média de XC_n; n o valor de $count(XC_n)$; $2,5$ a confiança em 99% na tabela de distribuição gaussiana (Z); e σ o desvio padrão de [Pearson 1913] dos valores XC_n.

Nas seções seguintes, são descritos o processo da configuração dos experimentos e coleta de dados, a análise dos resultados, e as considerações finais sobre os resultados.

5. Validação e Análise dos Resultados

A definição dos experimentos, aplicadas a três categorias de recursos distintos, em um ambiente de ataques por *stepping-stones* foi atribuída aos seguintes requisitos mínimos:

- a configuração de dois computadores para exercer as funções de cliente e servidor, e as demais máquinas reservadas para assumir o papel de *stepping-stone* no ataque;
- a interconectividade de todas as máquinas, uma a uma, entre si por conexões remotas por *SSH*;
- a execução do algoritmo em Python e o uso do analisador de protocolos *Wireshark* para captura do tráfego de pacotes na rede.

A primeira etapa do experimento consistiu da análise dos dados coletados na pesquisa em [Zhang 2014], para a investigação de um perfil de detecção. Para o desenvolvimento desta etapa, foram configuradas as seguintes máquinas: (i) um computador cliente, atribuído como atacante; (ii) um segundo como parte integrante da cadeia de *stepping-stones*; e (iii) uma terceira máquina, exercendo dupla função de servidor e vítima final nos ataques. A quarta máquina desempenhou a atribuição de *botnet* nos testes de invasão por *stepping-stones*. Enquanto às três primeiras máquinas estavam na rede do campus

 © 2017 Sociedade Brasileira de Computação

na Universidade de Houston, Texas, a quarta estava situada na cidade de Pittsburg, Pensilvânia.

A classificação dos dados, para a primeira etapa, sucedeu conforme a captura dos pacotes em cada uma das quatro máquinas da cadeia de *stepping-stones*. Os dados foram disponibilizados em planilhas do tipo CSV, e divididas entre longa e curtas conexões, contendo informações dos limites *inter-gaps*, em um total de quarenta experimentos.

A segunda etapa definiu a validação do algoritmo, ao investigar o comportamento do perfil de detecção definido na primeira etapa, por meio dos testes realizados em dados coletados da central de supercomputadores da Universidade de San Diego e disponibilizados pela . A base de dados do CAIDA[CAIDA 2016], dispõe de um tráfego de pacotes classificados entre dados coletados de usuários normais e pacotes capturados de conexões com ataques de negação de serviço, *botnets*, e exploração de vulnerabilidades.

A validação dos dados, para a base de dados definida por *Anonymized 2016 Internet Traces*, foi caracterizada pelo tráfego de dados passivo em tempo real com diferentes categorias de ataque. Os dados foram disponibilizados em formato *pcap*, classificados pela data da coleta e os arquivos continham entre 1 e $2,5\ GB$ de informações dos diferentes categorias de protocolo de rede.

A terceira etapa compreendeu da aplicação de um ataque no mundo real, por meio do uso de máquinas na nuvem da AWS. A elaboração de um modelo de ataques por *stepping-stones*, constitui de conexões formadas por cinco computadores e distribuídas da seguinte forma: (i) uma máquina cliente para o envio de ciberataques, localizada no Laboratório de Sistemas Distribuídos na UFCG; (ii) a configuração de três recursos da nuvem da AWS, localizadas respectivamente na Califórnia (EUA), Sidney (Austrália), e uma terceira em Singapura, atribuídas como parte integrante da cadeia de *stepping-stones*; e (iii) um servidor, atuando no papel de vítima, situado em Dublim, Irlanda.

A análise dos dados foi iniciada a partir do estabelecimento das conexões aos serviços da AWS. Um total de quarenta iterações de experimentos foram realizados, e para cada iteração, mil pacotes foram selecionados em tempo real, para definir um padrão dos dados, e os resultados foram salvos em arquivos de texto e em arquivos do tipo *pcap*, para fins de comprovação dos dados. A primeira etapa dos experimentos decorreu da análise dos dados coletados na pesquisa de [Zhang 2014].

Para às três etapas, houve uma necessidade de selecionar os pacotes relevantes, os que transportavam dados das conexões SSH, desconsiderando os demais.

O desenvolvimento do processo dos experimentos, consistiu em um modelo de fatores controláveis composto das informações de entrada, combinados aos fatores incontroláveis, determinados pelos limites *inter-gaps* discrepantes. O princípio básico do planejamento dos ensaios teve ênfase na possibilidade de replicação das iterações, sob as mesmas condições experimentais dos fatores aplicados no processo, conforme a ilustração da Figura 5.

Os fatores incontroláveis foram determinados pelos valores atípicos entre os quartis de um fluxo de informações para os limites *inter-gaps*. Os quartis representaram a separação dos limites *inter-gaps* divididos entre: (i) o quartil inferior Q_i que delimitou os 25% dos limites inter-gaps menores a Q_i e 75% dos limites *inter-gaps* maiores a Q_i; (ii)

© 2017 Sociedade Brasileira de Computação

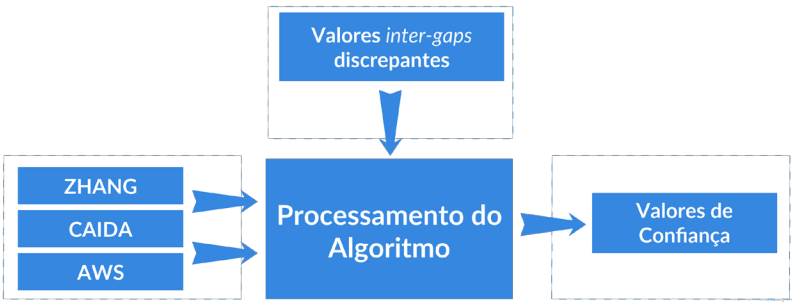


Figura 5. Planejamento dos Experimentos

o quartil central Q_c que correspondeu a 50% dos limites *inter-gaps* maiores e menores a Q_i e Q_s, respectivamente; (iii) e o quartil superior Q_s correspondente a 75% dos limites *inter-gaps* menores a Q_s e 25% dos limites *inter-gaps* maiores a Q_s.

A distribuição das informações para os valores discrepantes abrangeram os pontos extremos dos limites *inter-gaps* calculados em microssegundos (μs). Na primeira etapa dos experimentos, a frequência dos dados incomuns acima de Q_s foram contínuos em conexões estabelecidas por um intruso, e apresentaram uma variação nos quartis de $\Delta Q \cong 2,5\ \mu s$ valores atípicos por iteração. As curtas conexões ocasionalmente apresentaram dados discrepantes, retratando uma assiduidade entre um valor discrepante para cada três ensaios.

Para a segunda etapa, as distribuições dos fatores incontroláveis para amostras provenientes de ataques, foram contínuas, devido à presença de pontos discrepantes maiores que no quartil superior Q_s. A média individual para a presença de valores atípicos presentes na amostra foi de $\Delta Q \cong 18,33\ \mu s$. As amostras para arquivos provenientes de um tráfego de rede normal, apresentaram uma média de $\Delta Q \cong 1,33\ \mu s$ pontos discrepantes por rodada de experimento. A ausência de dados atípicos menores que o quartil inferior Q_i, para às duas etapas, determinou que os elementos discrepantes inferiores a Q_i não seriam considerados como fatores incontroláveis.

A variação dos valores discrepantes inferiores ΔQ_i e superiores ΔQ_s são apresentados pela Tabela 1, e detalham os resultados com os números máximos e mínimos dos valores de confiança, sobre às duas perspectivas de classificação das conexões.

Tabela 1. Resultado dos Valores Discrepantes em microssegundos (μs)

Base de Dados	Variação para Ataques	Variação para Usuários Normais
Zhang	$\Delta Q_i \cong 2,32$ e $\Delta Q_s \cong 7,84$	$\Delta Q_i \cong 0,36$ e $\Delta Q_s \cong 1,06$
CAIDA	$\Delta Q_i \cong 0,62$ e $\Delta Q_s \cong 2,01$	$\Delta Q_i \cong 0,003$ e $\Delta Q_s \cong 0,011$
AWS	$\Delta Q_i \cong 0,62$ e $\Delta Q_s \cong 7,84$	$\Delta Q_i \cong 0,43$ e $\Delta Q_s \cong 1,20$

Um ponto importante a ser destacado, na Tabela 1, foi referido ao número de valores discrepantes observados nos experimentos realizados na base de dados do CAIDA. O fator de detecção identificado nessa etapa, apresentou um número inferior, para às duas categorias de conexões, e divergiu dos resultados apresentados na base de dados de Zhang e AWS.

Os valores de confiança XC_n foram definidos como as variáveis de resposta do

 © 2017 Sociedade Brasileira de Computação

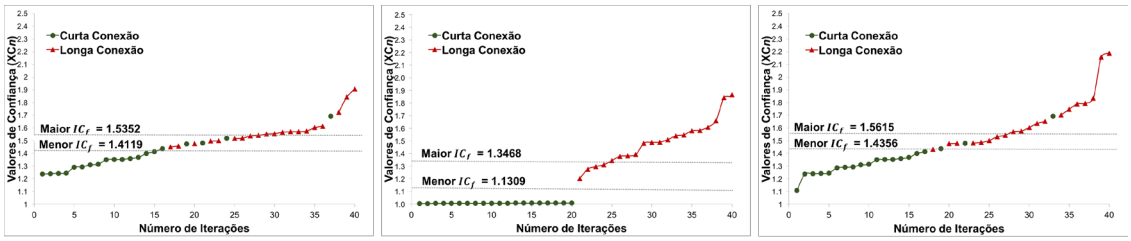
cálculo do risco relativo atribuído aos valores *inter-gaps* discrepantes. O propósito estabelecido aos valores de confiança XC_n foi determinar a estimativa de pontos aproximados de um verdadeiro valor, para definir um intervalo de confiança IC_f que delimite as informações entre ataque ou usuário real. O nível de confiança, para os resultados do algoritmo propostos neste trabalho, aplicados ao intervalo de confiança IC_f foi de 99% de confiança e uma incidência de erros em 1%. Na Tabela 2 é possível identificar os resultados para os valores mínimos e máximos do intervalo de confiança IC_f.

Tabela 2. Resultado dos Intervalo de Confiança

Base de Dados	Mínimo IC_f	Máximo IC_f
Zhang	1,4119 μs	1,5352 μs
CAIDA	1,1309 μs	1,3468 μs
AWS	1,4356 μs	1,5615 μs

Com base na Tabela 2, o intervalo de confiança para a base de dados CAIDA foi totalmente distinto da primeira e terceira etapa. Diante desse fato, foi possível inferir que a disposição anormal de valores discrepantes no segundo passo, pode indicar uma influência negativa nos resultados apresentados, e uma notável diferença na conclusão dos experimentos.

O intervalo de confiança IC_f foi determinado por meio da incidência de um risco relativo delimitado a partir dos valores de confiança XC_n. A delimitação do intervalo de confiança, a partir dos valores confiáveis, é ilustrada na Figura 6.



(a) Base de dados Zhang (b) Base de dados CAIDA (c) Base de dados AWS

Figura 6. Resultado do Intervalo de Confiança (IC_f) para as três etapas

Conforme apresentado na Figura 6, os resultados referentes a base de dados de Zhang, para curtas conexões foi de 95% e uma classificação de 19 verdadeiros positivos do 20 valores de confiança, e para longas conexões o resultado ficou com um índice de 100% de detecção, classificando os 20 verdadeiros negativos dos 20 valores de confiança XC_n. Os mesmos ensaios, aplicados a base de dados do CAIDA, viabilizou um índice de detecção de intrusos em 100% de acertos para às duas categorias de classificação entre as conexões. A aplicação do algoritmo a base de dados AWS, viabilizou a identificação de 19 dos 20 casos aplicados para longas conexões, e a detecção de 19 em 20 amostras, para conexões provenientes de um usuário real.

As métricas para avaliação de desempenho e mensuração dos resultados, foram calculadas com base na matriz de confusão, considerando as seguintes nomenclaturas: (i) curtas conexões positivas (CCP), como resultados provenientes de verdadeiros usuários;

© 2017 Sociedade Brasileira de Computação

(ii) as curtas conexões negativas (CCN), como valores de curtas conexões consideradas como ataque; (iii) as longas conexões positivas (LCP), considerados como conexões provenientes de intrusos; (iv) as longas conexões negativas (LCN), como valores provenientes de conexões estabelecidas por cibercriminosos que são classificadas como usuários reais; (v) a taxa de precisão (IPr), para determinar a taxa de acerto na classificação das conexões; (vi) a incidência de erros (IFa), calculada pela quantidade de alarmes falsos negativos produzidos pela classificação das longas conexões; e (vii) o índice de exatidão do algoritmo (IEt), determinado pelo número de longas e curtas conexões que foram corretamente classificadas. Na Tabela 3 é listado os resultados obtidos para cada etapa dos experimentos.

Tabela 3. Matriz de Confusão do resultado obtido para as três etapas

Base de Dados	CCP	CCN	LCP	LCN	IPr	IFa	IEt
Zhang	100%	0%	95%	5%	100%	5%	97,5%
CAIDA	100%	0%	100%	0%	100%	0%	100%
AWS	95%	5%	95%	5%	95%	5%	95%

Os resultados apresentados na Tabela 3, evidenciaram uma maior taxa de detecção, e, ao mesmo tempo, uma diminuição no índice de alerta falsos positivos. O índice de precisão dos resultados no trabalho proposto em [Zhang 2014] foi de 85% de detecção para longas conexões, em contrapartida, a aplicação do algoritmo a base de dados Zhang, nas mesmas condições de testes, apresentaram uma taxa de detecção de intrusos por *stepping-stones* em 97,5% na primeira etapa dos experimentos. Para a segunda, o processamento do algoritmo adotado para uma base de dados reconhecida internacionalmente no meio científico, retratou uma taxa de precisão de 100% para os casos testados. Na última etapa, o índice de precisão para um ambiente de máquinas configuradas em diferentes países foi de 95% de detecção de intrusos.

Em vista dos resultados obtidos neste trabalho, é possível evidenciar a superioridade do algoritmo proposto em relação às técnicas realizadas em [Ding and Huang 2011, Zhang 2014]. A redução na incidência de alarmes falso positivos comparados às técnicas anteriores indica a relevância do algoritmo na detecção de ataques do tipo *stepping-stones*.

6. Considerações Finais

O estudo acerca da detecção de intrusões por *stepping-stones* mostrou-se complexo desde as primeiras técnicas apresentadas em [Staniford-Chen and Heberlein 1995].

Na busca de atingir os objetivos direcionados na identificação de invasões por *stepping-stones*, o método de detecção proposto na presente pesquisa, aplicou o desenvolvimento de um algoritmo para classificar as conexões a partir da limitação dos limites *inter-gaps* aplicados a distribuição de informações com 99% de confiança. O diferencial da metodologia proposta em relação às técnicas existentes, está na identificação das tentativas de ataques a partir da análise dos dados coletados no computador da vítima, em tempo real de ataque, uma vez que os métodos anteriores desenvolveram soluções baseadas nas informações provenientes dos computadores intermediários envolvidos a uma cadeia de *stepping-stones*.

 © 2017 Sociedade Brasileira de Computação

A validação da solução proposta foi realizada por experimentos divididos em três etapas. A primeira etapa consistiu na aplicação do algoritmo aos dados coletados na pesquisa em [Zhang 2014], apresentando a proposta de criar um perfil de detecção a partir de informações já validadas e apresentou um índice de detecção de 97,5% para 5% de falsos positivos. O segundo passo, foi fundamentado na verificação do modelo aplicado a uma base de dados internacional reconhecida no meio científico e classificou as conexões a uma taxa de 100% de detecção para nenhum alerta falso positivo. A terceira etapa abrangeu uma incidência de 5% de falsos positivos e um índice de detecção em 95% para conexões configuradas em diferentes países.

Dentre as dificuldades encontradas, é destacado a alta concentração dos valores discrepantes ao longo da segunda fase dos experimentos e a influência negativa nos resultados para a base de dados do CAIDA. Outra dificuldade está relacionada na aplicação de um número reduzido de fatores incontroláveis produzidos pela ausência de métodos como *jitter* e *crossover* na organização dos experimentos.

É importante ressaltar que a finalidade estabelecida no atual trabalho é melhorar as soluções propostas na pesquisa de [Ding and Huang 2011, Zhang 2014] e contribuir cientificamente para a comunidade de pesquisadores na definição de novos conceitos e explicações a respeito da temática *stepping-stone*. Assim, as contribuições apontadas neste trabalho são evidenciadas na validação das pesquisas propostas por [Ding and Huang 2011, Zhang 2014], aplicadas ao algoritmo capaz de identificar intrusões por *stepping-stone* aplicados à base de dados para três arquiteturas de rede. Adicionalmente, a atribuição de diferentes recursos estatísticos, permite a aplicação do algoritmo proposto a qualquer ambiente que envolva ataques por *stepping-stones*.

Como trabalhos futuros, pretende-se incrementar uma solução que aplique as técnicas de *jitter*, *crossover* e perda de pacotes ao algoritmo proposto. Adicionalmente, incluir uma base de dados composta por informações pré-existentes dos padrões associados às longas e curtas conexões, viabilizando bloquear as intrusões por *stepping-stone* a partir da inferência comportamental das invasões. Por fim, implementar uma solução de processamento de *streams* ao algoritmo, para modelar um sistema de detecção de anomalias em tempo real.

7. Agradecimentos

Este trabalho foi parcialmente financiado pelo projeto EU-BRA SecureCloud (MCTI/RNP 3a Chamada Coordenada) e pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq).

Referências

CAIDA (2016). The CAIDA UCSD Anonymized Internet Traces 2016. `http://www.caida.org/data/passive/passive_2016_dataset.xml`.

Daud, A. Y., Ghazali, O., and Omar, M. N. (2015). Stepping-stone Detection Technique for Recognizing Legitimate and Attack Connections. In Jamaludin, Z., ChePa, N., Ishak, W. H. W., and Zaibon, S. B., editors, *5th International Conference on Computing and Informatics*, number 189, pages 440–446, Istanbul, Turkey. School of Com-

© 2017 Sociedade Brasileira de Computação

puting, University Utara Malaysia.

Ding, W. and Huang, S.-H. S. (2011). Detecting Intruders Using a Long Connection Chain to Connect to a Host. In *2011 IEEE International Conference on Advanced Information Networking and Applications*, pages 121–128, Biopoles, Singapure. IEEE.

Global, P. (2013). Na mira dos ataques cibernéticos - Pesquisa Global. Technical report, Ernest & Young, Rio de Janeiro, RJ.

Gosset, W. S. (1908). The Probable Error of a Mean. *Biometrika*, 6(1):1.

Herd, G. P. and Kriendler, J. (2013). *Understanding NATO in the 21st Century: Alliance Strategies, Security and Global Governance*. Contemporary Security Studies. Taylor & Francis, Abingdon, UK, 1 edition.

Huang, S.-H. S., Zhang, H., and Phay, M. (2016). Detecting Stepping-Stone Intruders by Identifying Crossover Packets in SSH Connections. In *2016 IEEE 30th International Conference on Advanced Information Networking and Applications (AINA)*, pages 1043–1050. IEEE.

Kumar, R. and Gupta, B. (2016). Neural Network Based Approach for Stepping Stone Detection under Delay and Chaff Perturbations. *Procedia Computer Science*, 85(Cms):155–165.

Kuo, Y.-W., Huang, S.-H. S., Ding, W., Kern, R., and Yang, J. (2010). Using Dynamic Programming Techniques to Detect Multi-hop Stepping-Stone Pairs in a Connection Chain. In *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, pages 198–205. IEEE.

Mittal, R., Lam, V. T., Dukkipati, N., Blem, E., Wassel, H., Ghobadi, M., Vahdat, A., Wang, Y., Wetherall, D., and Zats, D. (2015). Timely: Rtt-based congestion control for the datacenter. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, SIGCOMM '15, pages 537–550, New York, NY, USA. ACM.

Omar, M. N., Siregar, L., and Budiarto, R. (2008). Hybrid stepping stone detection method. In *2008 First International Conference on Distributed Framework and Applications*, pages 134–138. IEEE.

Pearson, K. (1913). On the Probable Error of a Coefficient of Correlation as found from a Fourfold Table. *Biometrika*, 9(1-2):22–27.

Staniford-Chen, S. and Heberlein, L. (1995). Holding intruders accountable on the Internet. In *Proceedings 1995 IEEE Symposium on Security and Privacy*, pages 39–49. IEEE Comput. Soc. Press.

Wu, H.-C. and Huang, S.-H. S. (2010). Neural networks-based detection of stepping-stone intrusion. *Expert Systems with Applications*, 37(2):1431–1437.

Zhang, H. (2014). *Detecting Network Intruders by Examining Packet Crossovers in Connections*. Dissertação (mestrado em ciências da computação), Dissertação (Mestrado em Ciências da Computação) – University of Houston, Texas.

Zhang, Y. and Paxson, V. (2000). Detecting stepping stones. In *Proceedings of the 9th Conference on USENIX Security Symposium - Volume 9*, SSYM'00, pages 13–13, Berkeley, CA, USA. USENIX Association.

 © 2017 Sociedade Brasileira de Computação