Um dataset de ataques Low Rate DDoS

Edson Barbosa de Souza, Anderson Fernandes Pereira dos Santos
Instituto Militar de Engenharia (IME)
Praça General Tibúrcio, 80, 22290-270, Praia Vermelha, Rio de Janeiro, RJ, Brasil.
**luizparente.felipe@eb.mil.br*

RESUMO: *Este artigo descreve os experimentos que foram realizados para gerar o dataset de ataques distribuídos de negação de serviço (DdoS) slowloris, sockstress. São comparadas algumas taxonomias de ataques DDoS.*

ABSTRACT: *This paper describes the experiments that were performed to generate the dataset of Low Rate Distributed Denial of Service (LR DDoS). The attacks in the dataset are slowloris and sockstress. The article compares some taxonomies of DDoS attacks.*

PALAVRAS-CHAVE: *Ataques Low Rate DDoS, slowloris, sockstress,LR DDoS.*

KEYWORDS: *LR DDoS attacks, slowloris, sockstress, dataset.*

1. Introdução

Os países, progressivamente, dependem de Tecnologia da Informação para prover os mais diversos serviços. Um ataque de negação de serviço (Denial of Service - DoS) torna um sistema indisponível aos seus usuários legítimos, por exemplo, inundando a rede (flooding) com tráfego malicioso, consumindo largura de banda de rede, ciclos de CPU, buffers de roteadores e outros recursos [1].

Um Ataque Distribuído de Negação de Serviço (DDoS: Distributed Denial of Service) é uma especialização de um ataque DoS e é costumeiramente executado por uma grande rede de dispositivos remotamente controlados que se encontram distribuídos geograficamente [2],[3].

O presente dataset foi desenvolvido para a pesquisa de ataques LR DDoS (Low Rate Distributed Denial of Service: Ataques Distribuídos de Negação de Serviço com baixas taxas de transmissão).

Há alguns datasets extensivamente empregados como CAIDA (Cooperative Association for Internet Data Analysis) DDoS Attack 2007, DARPA Intrusion Detection dataset e TUIDS (Tezpur University Intrusion Detection System) DDoS dataset. Nenhum destes contém os ataques LR DDoS slowloris ou sockstress.

Este trabalho está dividido em seis seções, além desta introdução: na seção 2 são explicadas as vulnerabilidades exploradas nos protocolos TCP e HTTP pelos ataques slowloris e sockstress, na seção 3 é apresentada uma taxonomia de ataques DDoS, na seção 4 é realizada a comparação dos trabalhos relacionados. A seção 5 descreve os experimentos e é realizada uma análise crítica sobre estes. Por fim, na Seção 6 são relatadas as conclusões do trabalho.

1.1. Contribuições

A principal contribuição da pesquisa é a descrição do *dataset* e análise crítica do experimento realizado para a sua obtenção. É apresentado o novo *dataset* de ataques *LR DDoS* desenvolvido, que vem preencher uma lacuna na área, já que outros *datasets* não apresentam os ataques *slowloris* e *sockstress*.

2. Os protocolos TCP e HTTP: Funcionamento, Vulnerabilidades e Ataques

Nesta seção são discutidas algumas vulnerabilidades no funcionamento dos protocolos TCP e HTTP.

2.1.O Protocolo TCP e o ataque *sockstress*

No protocolo TCP é assumido que, em geral, os usuários não são maliciosos, sendo esta uma vulnerabilidade normalmente explorada [4]. A definição formal do TCP foi realizada na RFC 793, com modificações na RFC 1122 e extensões na RFC 1323.

O cabeçalho do protocolo TCP tem tamanho variável. A estrutura básica possui valores bem definidos, como as portas de origem (16 *bits*) e de destino (16 *bits*), número de sequência (32 *bits*) e o número de reconhecimento (32 *bits*). O número de sequência e o número de reconhecimento (ACK) garantem a confiabilidade da transferência de dados.

O *handshake* de três vias do protocolo TCP pode ser explicado como segue [5]:

Ao cliente é atribuído um número de sequência inicial X e o envia em um segmento TCP, com o *bit* da *flag* SYN ativado e o *bit* da *flag* ACK desativado, para o servidor, especificando o endereço IP e a porta a que deseja se conectar. Por sua vez, o servidor responde com um segmento SYN-ACK que confirma X e envia o seu próprio número de sequência inicial, Y. Ao receber o segmento TCP SYN, o servidor processa o pedido de conexão e então aloca memória para armazenar informações sobre o cliente na tabela TCB (*transmission control block*). Por fim, o cliente confirma o número de sequência inicial escolhido pelo servidor no primeiro segmento de dados que enviar (ACK). O tamanho da janela de dados, que é um *buffer* que armazena os dados recebidos antes de enviar à camada de aplicação, foi negociado. O tamanho da janela

é definido como o número de bytes que o receptor é capaz de processar durante uma sessão do TCP, servindo, assim, como um controle de fluxo. O tamanho da janela do TCP é um parâmetro explorado em diversos ataques de negação de serviço, como será discutido na seção 4.2.

2.2.O Protocolo HTTP e o ataque *slowloris*

O protocolo HTTP (*Hypertext Transfer Protocol*), cuja versão mais recente é a 2.0, está definido na RFC 7540 e é compatível com a versão anterior 1.1, sendo decidido durante a fase de negociação qual das duas versões será utilizada [6]. Uma das diferenças entre as duas versões é que no HTTP 1.1 os cabeçalhos são enviados em modo texto e no HTTP 2.0 em modo binário, enviando os *headers* comprimidos com o algoritmo HPACK por padrão, diminuindo o volume de dados trafegados [6].

Na documentação do protocolo HTTP [7] é especificado que o servidor *web* aguarde a recepção completa da requisição HTTP (\r\n\r\n) antes de efetuar o processamento, o que permite alguns tipos de ataques específicos, como os *Slow* HTTP (*slowloris*, *slow body* e *slow read*). De acordo com a RFC 2616, seção 2.2, o fim de linha no cabeçalho do protocolo HTTP é marcado pelos caracteres "CR LF ou \r\n" e o fim do cabeçalho (início da mensagem) por "\r\n\r\n". Em uma requisição HTTP normal, o cliente envia HTTP GET ao servidor e este responde com o código 200, que significa sucesso na conexão, enviando em seguida os dados solicitados.

2.3. Ataques de Negação de Serviço

Um ataque *DoS* (*Denial of Service: Negação de Serviço*) tem como objetivo impedir que os equipamentos e serviços oferecidos pela rede sirvam aos usuários legítimos [8].

Já um ataque *DDoS*, que é uma especialização do ataque DoS e, frequentemente, utiliza equipamentos distribuídos geograficamente [9]. Os agentes de software denominados bots são instalados de forma ilegítima em um grande número de máquinas infectadas por um malware e são repetidamente empregados em ataques DDoS [10]. Os elementos que participam do ataque são chamados de zumbis e atuam sob as ordens de um nó mestre denominado botmaster. Uma botnet é definida como uma rede constituída de máquinas infectadas por um malware e que são preparadas para a execução de atividades ilegais em escala mundial [11],[12]. Uma botnet normalmente é utilizada para a geração do tráfego malicioso.

Os bots ou zumbis recebem as ordens do nó mestre chamado botmaster através de uma outra camada denominada *Comando e Controle* (C & C). A figura 1 exibe a configuração típica de uma botnet [11], as setas simbolizando as trocas de mensagens entre os bots e o Comando e Controle e o botmaster.

3.Taxonomias de Ataques DDoS

Segundo [13], os ataques *DDoS* podem ser classificados seguindo diversos critérios, sendo os principais:

Grau de automação (DA): manual, semi-automática e automática.

Mecanismo de comunicação (CM): direto, indireto.

- Tipo de vítima (VT): aplicação, *host*, recurso, rede, infraestrutura.
- Impacto na vítima (IV): disrupção, degradação.
- Possibilidade de Caracterização (PC): não caracterizável ou caracterizável.

3.1 Grau de automação (DA)

Em função do grau de automação, os ataques podem ser manuais, semi-automáticos, automáticos.

Em um ataque manual, o atacante executa um *scan* no equipamento da vítima, invade e comanda o ataque. Nos ataques semi-automáticos, a rede de ataque é constituída de um nó mestre e um nó escravo e as fases de recrutamento, exploração e infecção são automatizadas. Com base no mecanismo de comunicação entre os agentes, pode-se dividir em comunicação direta ou comunicação indireta. No mecanismo de comunicação direta, o mestre e os escravos precisam conhecer os seus respectivos endereços IP. Na comunicação indireta, um serviço de comunicação legítima é utilizada para sincronizar os agentes, como o IRC (*Internet Relay Chat)* e, atualmente, protocolos como o P2P (*peer-to-peer*) e o HTTP [14].

3.2. Tipo de vítima (VT)

A vítima do ataque pode ser uma aplicação, um *host espe-*

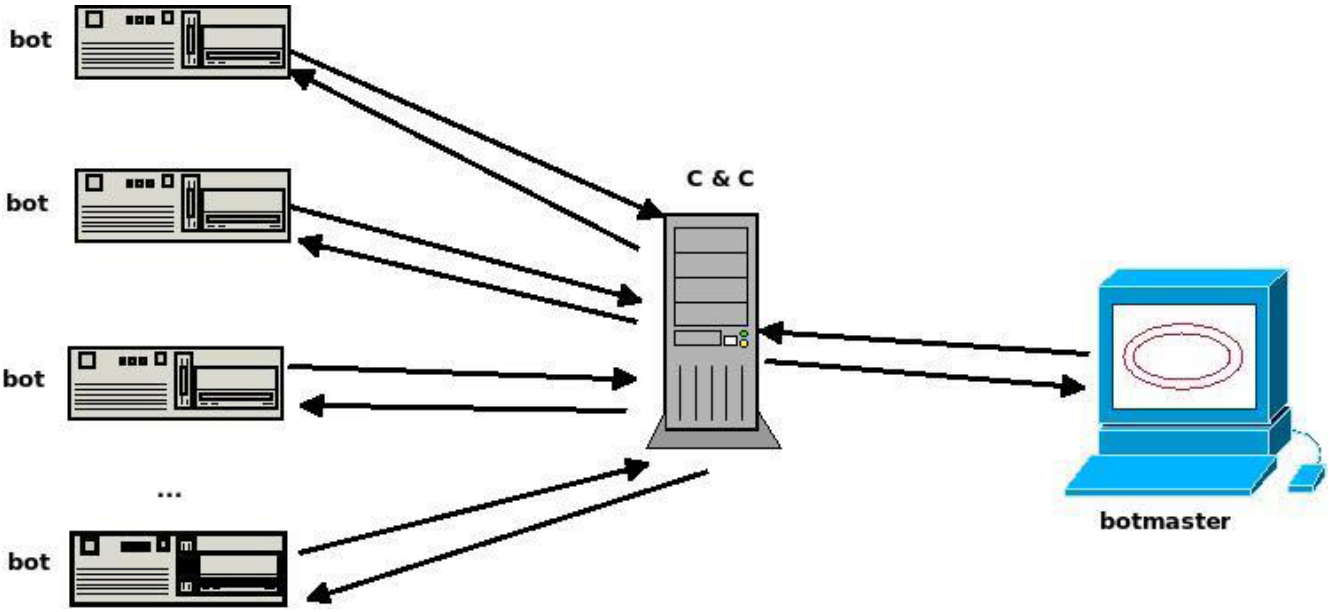

Fig. 1 – Configuração típica de uma botnet

cífico, um recurso crítico (como por exemplo, um roteador), a rede ou uma infraestrutura (como por exemplo, ataque a servidores DNS).

3.3. Impacto na vítima (IV)

Quando um ataque a um servidor interrompe totalmente os serviços oferecidos aos seus usuários legítimos, há uma disrupção. Por outro lado, é possível que, mesmo sob um ataque de negação de serviço, o servidor ainda atenda aos usuários, porém a uma taxa inferior a ideal. Neste caso, ocorre uma degradação de qualidade de serviço (*QoS*).

3.4. Possibilidade de Caracterização (PC)

Os ataques são classificados como caracterizáveis quando a sua identificação é possível pela inspeção dos cabeçalhos dos pacotes. Exemplos: TCP SYN *flooding*, UDP *flooding*, *smurf*.

Os ataques caracterizáveis subdividem-se em filtráveis e não filtráveis. Os ataques filtráveis são os que utilizam pacotes mal formados ou pacotes que não são necessários para a operação normal do equipamento alvo. Exemplos de ataques filtráveis a um servidor *web*: UDP *flooding e smurf.* Os ataques não filtráveis utilizam pacotes bem formatos e legítimos. Exemplos: HTTP *flooding*, ataques *slow* HTTP, *sockstress*.

Um ataque que usa uma estratégia refletida [15], Distributed Reflected Denial of Service (DRDoS: Ataque Distribuído e Refletido de Negação de Serviço), é caracterizado pelo uso de IP de origem falsificado (*spoofing*) e IP de destino como o IP de *broadcast* da rede da vítima. Ao invés de diretamente enviar o tráfego malicioso à vítima, nós intermediários, denominados de refletores, são utilizados. Os nós refletores, ao responderem às requisições, enviam os pacotes à vítima, ocasionando a negação de serviço. Exemplo clássico de ataque refletido é o *smurf* [15].

Outra taxonomia de ataques *DDoS* é descrita em [16]:

- Ataques baseados em volume (*Volume based*): os recursos computacionais da vítima (CPU, *buffers*, memória, rede, etc) são exauridos por um grande volume de trafego malicioso, o que faz usuários legítimos da aplicação serem preteridos, ocasionando a negação de serviço. O ataque ocorre na camada de rede, na de transporte ou na de aplicação. Exemplos: TCP SYN *flooding*, UDP *flooding*, *smurf.*
-
- Ataques na aplicação (*App based*): diferentes aplicações são o alvo do atacante para consumir os recursos da vítima. Exemplo: HTTP *flooding* (HTTP *GET* ou HTTP *POST*).
-
- Ataques *slow rate* ou *low rate* (SR/LR): são o tipo mais difícil de detectar, pois operam abaixo dos patamares de detecção e apresentam características similares ao tráfego legítimo [17]. Exemplos: ataques *slow* HTTP (como o *slowloris* e o RUDY*)*, *shrew, RoQ, sockstress.*

Os ataques LR DoS exploram protocolos como HTTP e HTTPS para enviar o tráfego a vítima, causando uma sobrecarga da CPU e tornando o servidor indisponível. O volume de tráfego necessário para tal negação de serviço é, em geral, menor que o empregado em ataques que utilizam inundação [18]. Exemplos de tais ataques são o *sockstress*, o *slowloris* (*slow* HTTP *headers*) e o RoQ.

3.5 Ataques LR *DDoS*

O ataque *sockstress,* que é um ataque a baixas taxas de transmissão, explora uma vulnerabilidade no protocolo TCP em relação ao tamanho da janela. O cliente malicioso estabelece a conexão TCP com tamanho de janela igual a zero e o servidor alvo continuamente enviará pacotes de descoberta do tamanho de janela a fim de aumentar seu valor e iniciar efetivamente a comunicação, mantendo a conexão aberta por tempo indeterminado e tendo esgotados seus recursos.

Os ataques do tipo *Slow* HTTP têm como alvo a camada de aplicação. Quando uma requisição HTTP é incompleta ou a taxa de transferência é muito baixa, o servidor *web* mantém a conexão aberta aguardando o restante dos dados [19]. Há três tipos de técnicas utilizadas pelos ataques *Slow* HTTP [19]:

1. Enviar o cabeçalho da requisição lentamente;
2. Ler a resposta lentamente; e
3. Enviar o corpo da resposta da requisição lentamente.

O primeiro tipo é denominado *slowloris* (ou *Slow* HTTP *Headers* ou *Slow* HTTP GET); o segundo é o *Slow Read* e o terceiro *Slow body* ou RUDY.

No ataque *slowloris*, o atacante envia os cabeçalhos HTTP sem os caracteres indicativos de seu final (r\n\r\n), repetidamente de modo a manter a conexão ativa. O servidor mantém a conexão aberta aguardando o restante do cabeçalho, que o atacante nunca enviará por completo. Sem precisar de muita largura de banda, um atacante pode abrir muitas conexões e sobrecarregar o servidor *web*.

Na figura 2, que exibe o cabeçalho HTTP em uma conexão normal, nota-se o "\r\n\r\n" indicando o fim do *header* e início dos dados requisitados. A figura 3 mostra o cabeçalho HTTP durante um ataque *slowloris*. Observa-se apenas um "\r\n" de forma a manter a conexão ativa.

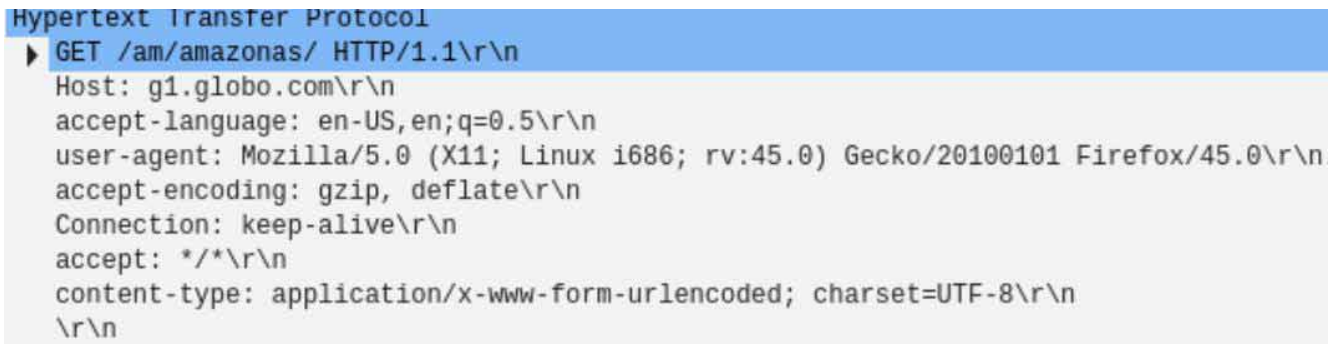

Fig. 2– cabeçalho em uma conexão HTTP normal

Fig. 3– cabeçalho HTTP em um ataque slowloris

No ataque *Slow Read,* o *handshake* de três vias é completado de forma legítima mas o atacante define um valor de tamanho de janela muito pequeno, como por exemplo 700 *bytes*, ao enviar o segmento ACK [20]. Desta forma, a resposta é recebida do servidor de forma lenta. Quando o tamanho da janela está próximo a zero, o servidor para de enviar dados e mantém a conexão ativa. O servidor é indisponibilizado ao serem efetuadas um número de conexões concorrentes suficientes para esgotar seus recursos. Não existem contramedidas conhecidas contra o ataque *slow read* [20].

O ataque *Slow body* ou RUDY é caracterizado pelo envio do HTTP Post header completo mas com o campo *content-length* (tamanho da página em *bytes*) maior do que o seu tamanho real e enviado lentamente (por exemplo, 1 *byte*/min) de forma a manter o servidor ocupado [21].

4. Trabalhos Relacionados

Dentre os principais *datasets* largamente utilizados em pesquisas de ataques distribuídos de negação de serviço, destacam-se o CAIDA *DDoS Attack* 2007, o DEFCON-10 *dataset, o KDD Cup 1999 dataset, o TUIDS DDoS dataset,* o DARPA *Intrusion Detection dataset*, e o NSL-KDD *dataset* [1],[22].

No CAIDA *DDoS Attack* 2007 *dataset*, o *payload* dos pacotes foi totalmente removido e os cabeçalhos estão disponíveis até a camada de transporte [1].

O DEFCON-10 *dataset* foi criado em 2002. O seu tráfego foi obtido na competição *Capture The Flag* (CTF), sendo, pois, muito diferente de um tráfego realístico (contém apenas conexões maliciosas como *port scans* e varreduras) sem um tráfego de fundo normal; é muito utilizado para a avaliação de IDS [22].

O *dataset Knowledge Discovery and Data Mining* (KDD Cup 1999) é muito utilizado em pesquisas de *malwares* mas não é adequado para a avaliação de detecção de ataques *DDoS* ou *flash events.* Alguns problemas do *dataset* são a grande quantidade de dados redundantes, em torno de 75%, dados de ataques falsos e antiquados para avaliação [1]. O *dataset* KDD Cup 1999 não contém os tipos de ataques mais recentes [23].

O TUIDS *DDoS dataset* foi desenvolvido no final de 2015, no entanto, muitos ataques estão desatualizados e exploram apenas sistemas operacionais antigos como o Windows 95 ou Windows NT 4.0 [23].

O DARPA *Intrusion Detection dataset*, sendo o DARPA *Intrusion Detection dataset* 2000 o mais utilizado, não abrange novas técnicas de ataque. Contém registros de auditoria do sistema operacional Windows NT e ataques *DoS* como *neptune*, *smurf* e *ping of death*. É ainda útil para testes de IDS [24],[25].

O NSL-KDD *dataset* foi desenvolvido para resolver alguns problemas inerentes ao KDD Cup 1999 *dataset.* Assim, sendo, não contém os dados redundantes.

A pesquisa de [26] apresenta um *dataset* contendo o ataque slowloris mas não os arquivos de captura (*dump*) completos. Os endereços de IP foram removidos, e o tráfego é rotulado com "1", caso seja malicioso e "-1" em tráfego legítimo, em formato csv. Em [27], o *dataset* contém os ataques RUDY e *slowloris* mas não está disponível publicamente na página do projeto.

A comparação entre os *datasets* está resumida na tabela 1.

Tab 1: Comparação de datasets com ataques DDoS

	Arquivo de captura completo	Ataques LR DDoS slowloris/ sockstress	Tráfego Realístico	Observações
CAIDA DDoS	NÃO	NÃO	SIM	Sem payload. Algumas flags e informações de protocolos removidos
DARPA 2000	SIM	NÃO	NÃO	-
KDD-99	SIM	NÃO	NÃO	Tráfego de fundo e ataques não estão bem identificados.
TUIDS DDoS	SIM	NÃO	SIM	-
DEFCON-10	SIM	NÃO	NÃO	Apenas tráfego intrusivo.
NSL-KDD	SIM	NÃO	NÃO	-

5. Descrição do Experimento

Os servidores *web* mais utilizados são o Apache e o IIS [28], [29]. Neste trabalho foi utilizada a versão 2.4.10 do Apache.

Foi usado um laboratório cuja configuração é a seguinte:

- 11 equipamentos *Desktops;*
- Processador Intel *core* i7-4770S;
- 8GB de memória RAM;
- Disco Rígido de 1 TB;
- Sistema Operacional Ubuntu 14.01 64 *bits;*
- Rede LAN cabeada;
- *Switch* Catalyst 2960.

Cada núcleo do processador possui *cache* de 8192 KB e 800 MHZ.

Foram empregadas 11 máquinas físicas, sendo 10 atacantes, cada uma com 7 máquinas virtuais Debian Linux versão 8.7 32 *bits*, e gerando tráfegos *sockstress* e *slowloris*. Em um dos equipamentos, foi instalado o servidor *web* virtual com Debian Linux versão 8.7 32 *bits* instalado e o tráfego de fundo gerado com o wget[1] configurado no crontab[2] do Linux. O tráfego de fundo foi gerado com a ferramenta httpmon, que gera tráfego HTTP com intervalo entre duas requisições consecutivas seguindo uma distribuição exponencial.

As placas de rede das máquinas virtuais foram configuradas em modo *bridge* com a faixa de IP 192.168.91.X, máscara de rede 255.255.255.0, sendo 192.168.91.5 o IP do servidor *web*.

Cada uma das máquinas virtuais atacantes com o sistema Debian foi configurada com 600 MB de memória RAM e o servidor web virtual, 1.2 GB de memória RAM e 100 GB de HD. As informações referentes às configurações das máquinas virtuais estão descritas na tabela 2.

A ferramenta *opensource slowhttptest*, que vem instalada por padrão em muitas distribuições Linux, foi utilizada para gerar os ataques. Ela possui diversos módulos que permitem gerar os tipos de ataques *Slow* HTTP como *slowlo-*

[1] Utilitário para download não-interativo

[2] Agendador de Tarefas

ris, *slow read* e *slow body,* sendo muito utilizada para testar servidores *web* a fim de buscar vulnerabilidades a ataques *DDoS* ou simplesmente checar quantas conexões simultâneas o servidor pode gerenciar. Pode-se, inclusive, iniciar um ataque utilizando o slowhttptest a partir de um *smartphone*.

Embora os ataques estivessem programados para iniciar em um horário fixo, estes duravam no máximo 240 segundos e, era aguardado um tempo aleatório entre 180 segundos e 250 segundos para começarem novamente.

O tráfego foi coletado no formato PCAP[3] na máquina física contendo o servidor *web*. Os relógios internos dos equipamentos foram sincronizados com o servidor do **NTP.br usando** o protocolo NTP.

Tab 2: Configurações das máquinas virtuais

	Servidor web	***Atacantes***
Memória RAM	1.2 GB	600 MB
HD	100 GB	100 GB
Faixa de IP	192.168.91.5	192.168.91.1/24
S.O.	Debian 8.7 32 *bits*	Debian 8.7 32 *bits*
Total	1 servidor Apache	70 atacantes

O tráfego de rede capturado no formato PCAP foi constituído de:

- tráfego de fundo (TF), simulação de acesso a sites web, e
- tráfego de ataque (TA), que é constituído dos ataques *sockstress*, *slowloris*.

O tráfego coletado está distribuído conforme a tabela 3.

Tab 3: Distribuição do tráfego capturado no experimento

Horário	Conteúdo
07:50:00	Apenas Tráfego de Fundo
08:50:00	Início do Ataque *sockstress*
09:30:00	Fim do Ataque *sockstress*
09:40:00	Início do Ataque *slowloris*
10:20:00	Fim do Ataque *slowloris*

O *dataset* contém, além dos *dumps* em formato PCAP, os seguintes arquivos:

- *logs* do servidor Apache (*access*.log e *error*.log);
- as informações do estado da memória (total, livre, usada, *buffers* e *cache*) e CPU
- durante todo o experimento (formatos txt , XML); e
- os metadados dos ataques em formato XML.

Os valores de utilização de CPU, memória e rede (dados recebidos e transmitidos) do servidor *web* foram monitorados antes e durante os ataques com a ferramenta `Sysstat`. Em relação a CPU, são armazenados o tempo do usuário (us), o tempo de CPU dedicado aos códigos que não são do *kernel*, tempo de sistema (sy), que é o tempo dedicado aos códigos do *kernel*, tempo ocioso do sistema (id) e o tempo de espera por operações de entrada e saída (wa).

As medidas utilizadas para avaliar o impacto dos ataques no servidor *web* são a média do percentual de CPU ociosa (CPU i), o seu desvio padrão (CPU sd), a média de memória utilizada em KB (Mem), o seu desvio padrão (Mem sd), a média do total de KB recebidos por segundo (rxkB), o seu desvio padrão (rxkB sd), a média do total de KB transmitidos por segundo (txkB) e o seu desvio padrão (txkB sd). Os resultados são apresentados nas tabelas 4 e 5. As análises estão na seção 5.1.

Tab 4: Estatísticas 1

	CPU i	CPU sd	Mem	Mem sd
Sem Ataques	98%	0,90	285642	73700
Slowloris	93%	1,32	363621	25716
Sockstress	96%	1,65	343697	30454

Tab 5: Estatísticas 2

	rxkB	rxkB sd	txkB	txkB sd
Sem Ataques	10	2,03	8,54	1,88
Slowloris	108	21,53	70,73	13,77
Sockstress	8893	4600	3765	3518

5.1 Análise Crítica do Experimento

Será feita a comparação entre o comportamento de pacotes antes e depois da ocorrência de um ataque *LR DDoS* [30].

Antes do início do ataque *slowloris*, o arquivo *access*.log contido em /var/log/apache2 exibia o código 200 do HTTP, o que apontava o sucesso na requisição ao servidor. Após o início do ataque, o código 408 foi a resposta do servidor, ou seja, ocorreu *timeout* das requisições. Após o fim do ataque, o código de resposta do HTTP voltou a ser o 200.

Observando os números apresentados nas tabelas 4 e 5, percebe-se que o ataque *slowloris* foi efetivo em aumentar a utilização dos recursos do servidor *web*. A máquina virtual que continha o servidor *web* Apache teve um aumento de 5% na média de utilização de CPU, 27,3% de aumento na média de utilização de memória, a média de pacotes recebidos aumentou de 10,8 vezes e a média de pacotes enviados aumentou de 8,3 vezes durante o ataque *slowloris*.

Ao observar o *dump* do ataque *slowloris*, nota-se a regularidade dos tamanhos das janelas do TCP (os valores são sempre os mesmos), conforme a figura 7, comparando-se com os tamanhos das janelas no tráfego normal (figura 6). A figura 8 mostra o reduzido tamanho de janela durante o ataque *sockstress*.

Contando-se o número de conexões diferentes abertas por IP 192.168.91.5 (servidor *web*) na porta 80 por segundo, durante o ataque *slowloris*, em cada *time slice* de 1 minuto, tem-se uma média de 152 conexões por segundo, conforme a figura 4. Análise similar pode ser efetuada com o ataque *sockstress*, resultando em 52 conexões por segundo, conforme a figura 5.

Uma instalação padrão do Apache permite no máximo 150 conexões simultâneas (variável *MaxRequestWorkers*), conforme a sua documentação oficial.

6. Conclusão

Conforme os experimentos demonstraram, o ataque *slowloris* é efetivo para consumir os recursos de um servidor web Apache, especialmente a memória, e torná-lo indisponível. Este trabalho apresentou o novo *dataset* de ataques LR *DDoS* desenvolvido, que vem preencher uma lacuna na área, já que outros *datasets* não apresentam os ataques *slowloris* e *sockstress*.

[3] Packet capture: captura de pacotes

Com 70 máquinas virtuais atacantes, foram geradas em média 152 conexões por segundo durante o ataque *slowloris* e o servidor Apache teve um incremento de 5% na utilização de CPU, 27,3% em memoria, 10,8 vezes em pacotes recebidos e 8,3 vezes em pacotes enviados.

Como os ataques do tipo *slow* HTTP geram um tráfego quase idêntico a um legítimo, sua detecção torna-se difícil para *firewalls*.

Referências Bibliográficas

[1] Behal, S.; Kumar, K. Trends in Validation of DDoS Research. Procedia Computer Science, v. 85, n. Cms, p. 7–15, 2016.

[2] Santos, A. F P., Renato S. Silva. Detecção de Ataques DDoS com Gráficos de Controle e Bases de Regras Nebulosas,Rio de Janeiro,2009.

[3] Santos, A. F. P., and Renato S. Silva.Detecting bandwidth ddos attack with control charts. Networks, 2007. ICON 2007. 15th IEEE International Conference on. IEEE, 2007.

[4] Kuzmanovic, A.; Knightly, E. W. Low-rate TCP-targeted denial of service attacks: the shrew vs. the mice and elephants. Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications. Anais...2003.

[5] Tanenbaum, Andrew S. Redes de computadores. São Paulo: Pearson Pretice Hall,2011.

[6] Dimitrova, Biljana; Mileva, Aleksandra. Steganography of Hypertext Transfer Protocol Version 2 (HTTP/2). Journal of Computer and Communications, v. 5, p. 98-111, 2017.

[7] IETF. RFC 7540 - Hypertext Transfer Protocol version 2 (HTTP/2). Disponível em: <https://tools.ietf.org/html/rfc7540>. Acesso em: 01 jun. 2017.

[8] Dittrich, D. et al. Internet Denial of Service: Attack and Defense Mechanisms. [s.l.] Pearson Education, 2004.

[9] Zargar, S. T.; Joshi, J.; Tipper, D. A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. IEEE Communications Surveys & Tutorials, v. 15, n. 4, p. 2046–2069, 2013.

[10] Rai, Ankur; Challa, Rama Krishna. Survey on Recent DDoS Mitigation Techniques and Comparative Analysis. Computational Intelligence & Communication Technology (CICT), Ghaziabad, India, p. 96-101, fev. 2016.

[11] Silva, S. S. et al. Botnets: A survey. Computer Networks, [S.L], v. 57, n. 2, p. 378-403, fev. 2013.

[12] Zlomislić, Vinko; Fertalj, Krešimir; Sruk, Vlado. Denial of service attacks, defences and research challenges. Cluster Computing, New York, v. 20, n. 1, p. 661-671, jan. 2017.

[13] Mirkovic, Jelena; Reiher, Peter. A Taxonomy of DDoS Attack and DDoS Defense Mechanisms. ACM SIGCOMM Computer Communications Review, New York, NY, USA, v. 34, n. 2, p. 39-53, abr. 2004.

[14] Sengar, Bhan; Padmavathi, B.. A Survey on Botnet: Behavior, Life Cycle, Detection and Prevention. International Journal of Computer Technology and Applications, [S.L], v. 10, n. 9, p. 577-566, mar. 2017.

[15] Alieyan, K. et al. An overview of DDoS attacks based on DNS. Information and Communication Technology Convergence (ICTC), [S.L], p. 276-280, out. 2016.

[16] Rai, Ankur; Challa, Rama Krishna. Survey on Recent DDoS Mitigation Techniques and Comparative Analysis. Computational Intelligence & Communication Technology (CICT), [S.L], p. 96-101, ago. 2016.

[17] Kuzmanovic, Aleksandar; Knightly, Edward W.. Low-rate TCP--targeted denial of service attacks and counter strategies. IEEE/ACM Transactions on Networking, USA, v. 14, n. 4, p. 683-696, ago. 2006.

[18] Bhattacharyya, D. K.; Kalita, J. K. DDoS attacks. [s.l.] CRC Press, 2016.

[19] J. P. et al. Analysis of Slow Read DoS Attack and Countermeasures on Web servers. International Journal of Cyber-Security and Digital Forensics (IJCSDF), Hong Kong, v. 4, n. 2, p. 339-353, jan. 2015.

[20] Tayama, Shunsuke, and Hidema Tanaka. “Analysis of Slow Read DoS Attack and Communication Environment.” International Conference on Mobile and Wireless Technology. Springer, Singapore, 2017.

[21] Jazi, Hossein Hadian et al. Detecting HTTP-based application layer DoS attacks on web servers in the presence of sampling. Computer Networks, v. 121, p. 25-36, 2017.

[22] Shiravi, A. et al. Toward developing a systematic approach to generate benchmark datasets for intrusion detection. Computers and Security, v. 31, n. 3, p. 357–374, 2012.

[23] Dromard, J. et al. Online Network Traffic Characterization Deliverable Experimental evaluation of algorithms for online network. 2017.

[24] Shah, V. M.; Agarwal, A. K. Reliable Alert Fusion of Multiple Intrusion Detection Systems. v. 19, n. 2, p. 182–192, 2017.

[25] 2000 DARPA INTRUSION DETECTION SCENARIO SPECIFIC DATA SETS. DARPA. Disponível em: <http://www.ll.mit.edu/ideval/data/2000data.html>. Acesso em: 05 mai. 2017.

[26] Kumar, Raneel, Sunil Pranit Lal, and Alok Sharma. “Detecting Denial of Service Attacks in the Cloud.” Dependable, Autonomic and Secure Computing, 14th Intl Conf on Pervasive Intelligence and Computing, 2nd Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress (DASC/PiCom/DataCom/CyberSciTech), 2016 IEEE 14th Intl C. IEEE, 2016.

[27] Jazi, Hossein Hadian, Hugo Gonzalez, Natalia Stakhanova, and Ali A. Ghorbani. “Detecting HTTP-based application layer DoS attacks on web servers in the presence of sampling.” Computer Networks 121 (2017): 25-36.

[28] NETCRAFT. July 2017 Web Server Survey. Disponível em: <https://news.netcraft.com/archives/2017/07/20/july-2017-web--server-survey.html>. Acesso em: 17 out 2017.

[29] Tripathi, Nikhil, Neminath Hubballi, and Yogendra Singh. “How Secure are Web Servers? An Empirical Study of Slow HTTP DoS Attacks and Detection.” Availability, Reliability and Security (ARES), 2016 11th International Conference on. IEEE, 2016.

[30] Santos, A. F. P.; Silva, R. S. Detectando Ataques de Negação de Serviço, XXX Congresso Nacional de Matemática Aplicada e Computacional, 2007, Florianópolis, Brasil.

ANEXOS

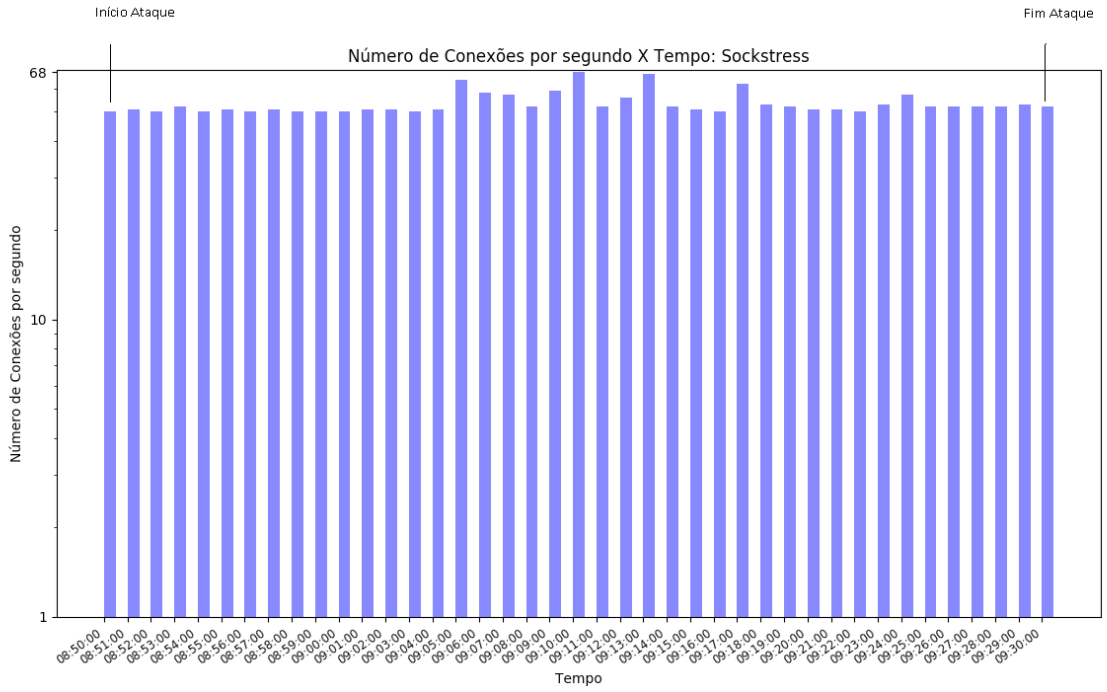

Fig. 4–Número de Conexões/segundo com ataque Sockstress

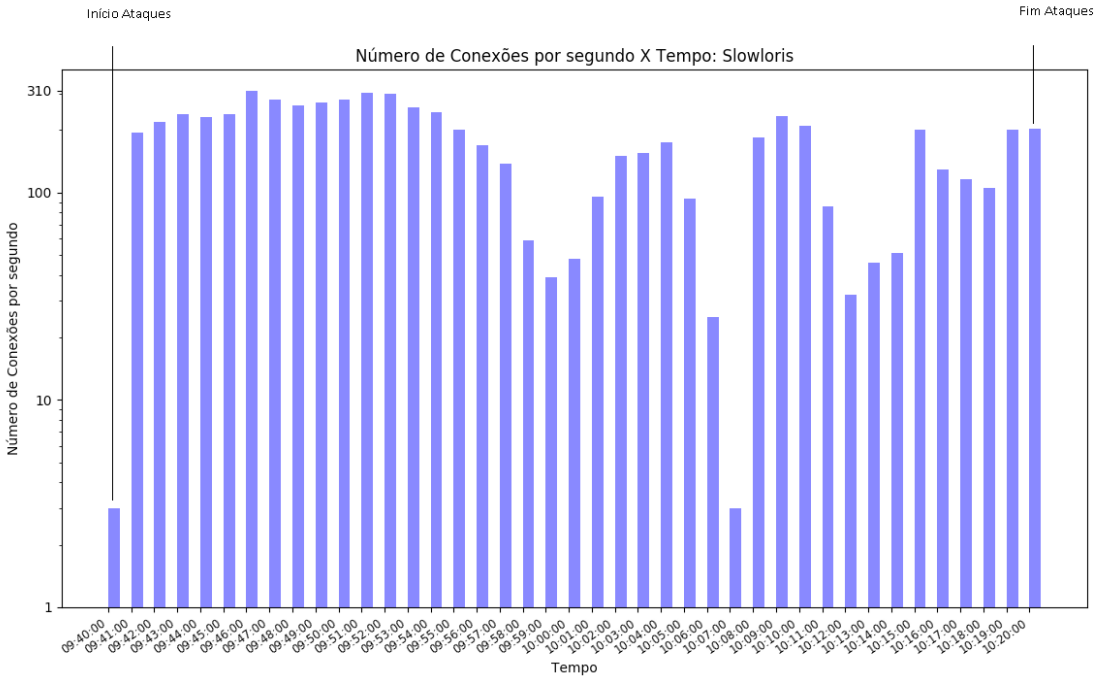

Fig.5– Número de Conexões/segundo com ataque Slowloris

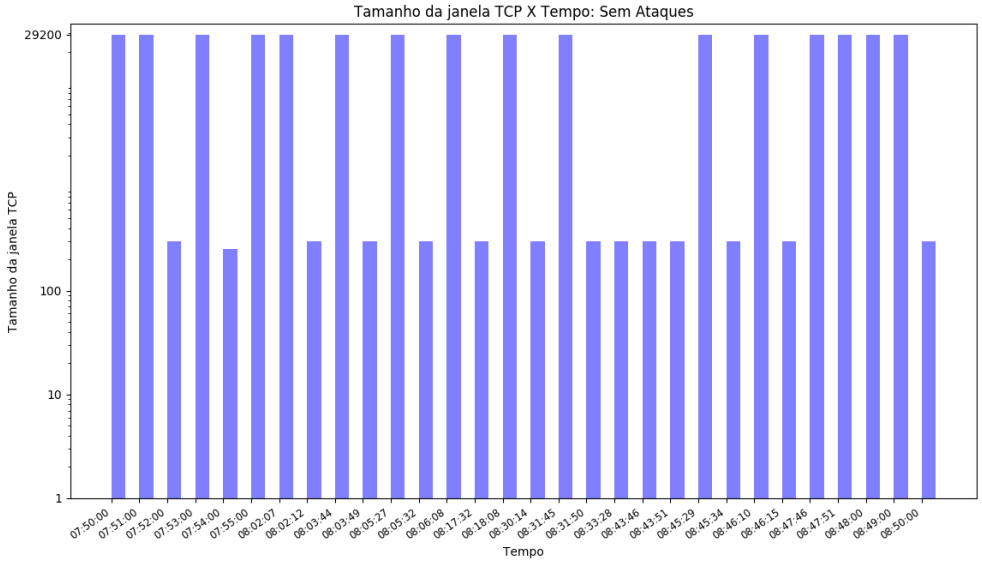

Fig.6- Tamanho da janela X tempo (sem Ataques)

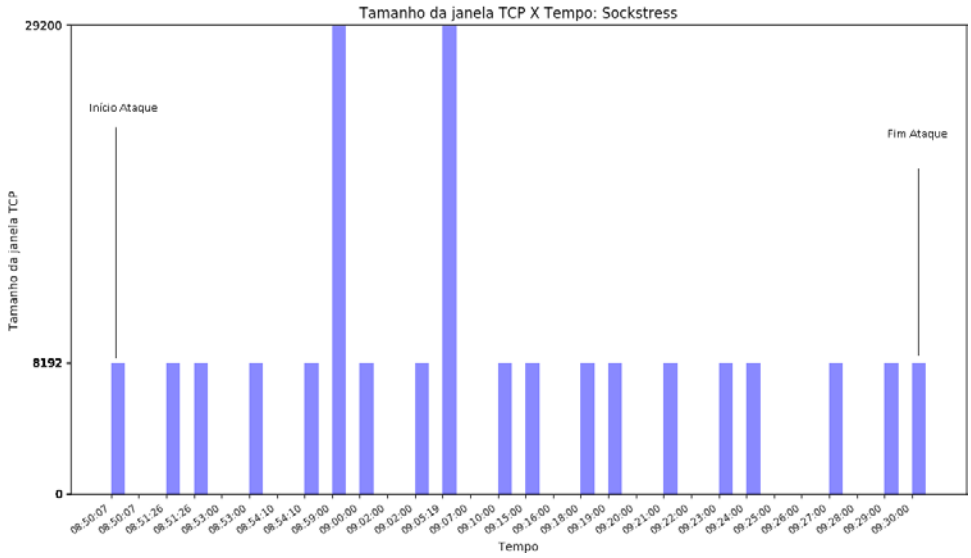

Fig.7- Tamanho da janela X tempo (sockstress)

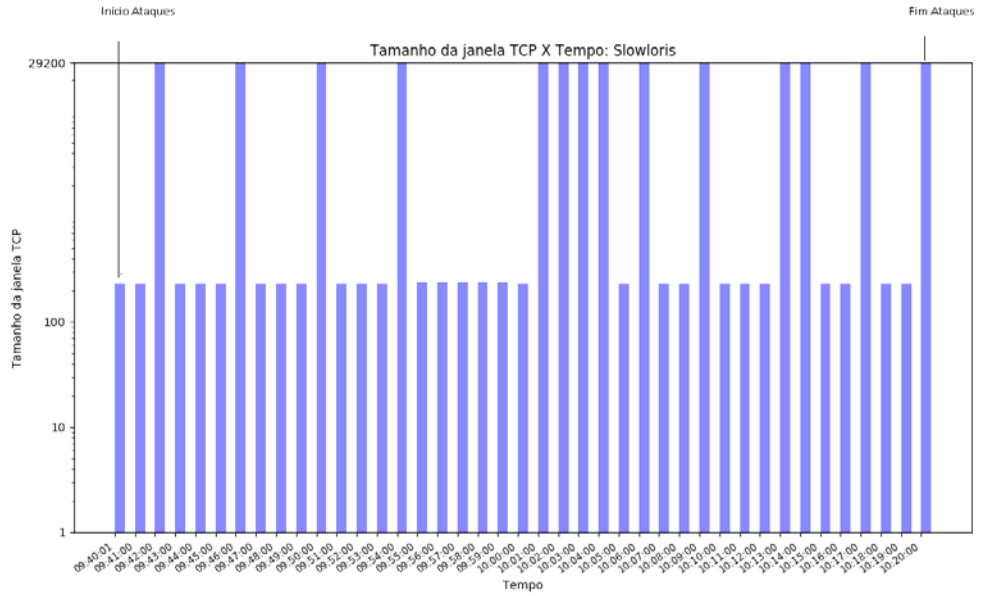

Fig.8- Tamanho da janela X tempo (slowloris)