Universidade Federal Fluminense

HENRIQUE DORNEL DA SILVA

Detecção de Ataques de Negação de Serviço em Redes de Computadores Através de Previsões por Séries Temporais

VOLTA REDONDA

2018

HENRIQUE DORNEL DA SILVA

Detecção de Ataques de Negação de Serviço em Redes de Computadores Através de Previsões por Séries Temporais

Dissertação apresentada ao Programa de Pós-graduação em Modelagem Computacional em Ciência e Tecnologia da Universidade Federal Fluminense, como requisito parcial para obtenção do título de Mestre em Modelagem Computacional em Ciência e Tecnologia. Área de Concentração: Modelagem Computacional.

Orientador:

Eliane da Silva Christo

Coorientador:

Kelly Alonso Costa

Universidade Federal Fluminense

VOLTA REDONDA

2018

Ficha catalográfica automática - SDC/BEM

S586d Silva, Henrique Dornel da
Detecção de Ataques de Negação de Serviço em Redes de Computadores Através de Previsões por Séries Temporais / Henrique Dornel da Silva ; Eliane da Silva Christo, orientadora ; Kelly Alonso Costa, coorientadora. Volta Redonda, 2018.
94 f. : il.

Dissertação (mestrado)-Universidade Federal Fluminense, Volta Redonda, 2018.

DOI: http://dx.doi.org/10.22409/PPG-MCCT.2018.m.12454492733

1. Rede de comunicação de computadores. 2. Segurança da informação. 3. Série temporal. 4. Produção intelectual. I. Título II. Christo,Eliane da Silva, orientadora. III. Costa, Kelly Alonso, coorientadora. IV. Universidade Federal Fluminense. Escola de Engenharia Industrial e Metalúrgica de Volta Redonda.

CDD -

Bibliotecária responsável: Helania Oliveira Madureira - CRB7/4250

Detecção de Ataques de Negação de Serviço em Redes de Computadores Através de Previsões por Séries Temporais

Henrique Dornel da Silva

Dissertação apresentada ao Programa de Pós-graduação em Modelagem Computacional em Ciência e Tecnologia da Universidade Federal Fluminense, como requisito parcial para obtenção do título de Mestre em Modelagem Computacional em Ciência e Tecnologia. Área de Concentração: Modelagem Computacional.

Aprovada por:

Prof. Eliane da Silva Christo, D.Sc. / MCCT-UFF
(Orientador)

Prof. Kelly Alonso Costa, D.Sc. / PPGEP-UFF
(Coorientador)

Prof. Tiago Araújo Neves, D.Sc. / MCCT-UFF

Prof. Cecilia Toledo Hernández, D.Sc. / MCCT-UFF

Prof. Gustavo Benitez Alvarez, D.Sc. / MCCT-UFF

Prof. Rafael Alves Bonfim de Queiroz, D.Sc. / DCC-UFJF

Volta Redonda, Agosto de 2018.

"Stay hungry, stay foolish."
Steve Jobs

"Se queres prever o futuro, estuda o passado."
Confucio

Dedico este trabalho ao meu tio Chico. Vá em paz, garotão!

Agradecimentos

Agradeço primeiramente ao meu pai Sérgio e à minha mãe Goretti, que sempre me incentivaram a continuar sempre estudando, evoluindo e acreditando que eu poderia que ir longe com meu potencial. Sou muito grato pelo amor, pela cobrança, pelo grande apoio e por terem sempre me ajudado a fazer as melhores escolhas na vida. Vocês são os melhores pais que eu poderia ter! Amo vocês!

Ao meu irmão Leonardo, por ser essa pessoa que eu amo e tão importante para mim. Apesar das dificuldades, você está se tornando um vitorioso! No momento mais crítico para a realização deste trabalho, você se tornou uma grande inspiração para mim, pela sua garra e dedicação!

À minha namorada Vanessa, uma das pessoas mais marcantes na minha vida, e que sempre acreditou que eu conseguiria chegar até aqui. Ao longo do último ano, todo esse amor, carinho, compreensão e cumplicidade foram muito importantes para a realização e conclusão deste trabalho. Eu te amo!

Aos meus amigos Leo e Ingrid, que foram fundamentais para a realização dessa conquista. Apesar da distância, vocês foram e sempre vão ser minha maior inspiração na vida acadêmica. Tudo o que aprendemos e vivenciamos juntos, seja na UFF ou na vida, contribuiu enormemente para que hoje eu pudesse dar o meu melhor e concretizar este trabalho. #PelaTour!

Ao meu grande amigo Vitor, que faz parte da minha vida há mais de duas décadas. Sou grato por tudo que vivemos e compartilhamos na escola, na faculdade, no trabalho e na vida, e por cada conversa, cada risada e cada conselho. Quando eu tive dificuldades na etapa mais importante deste trabalho, o seu apoio e incentivo foram um grande diferencial para que eu nunca desistisse e pudesse vencer esse desafio!

A todos os professores que eu tive ao longo da minha vida, desde a escola até a pós-graduação. Cada ensinamento foi fundamental para a construção do meu conhecimento e da minha vontade de sempre aprender mais. Agradeço especialmente aos professores da UFF (do ICEx e do MCCT), por todo o aprendizado e inspiração.

À minha orientadora Eliane, por ter enxergado uma ideia que veio a se tornar o presente trabalho de dissertação, e por ter me convidado a ser seu orientando. Sou muito grato por toda a paciência, suporte e incentivo, principalmente nos momentos mais difíceis.

À minha coorientadora Kelly, pela enorme contribuição para este trabalho. Agradeço a cada sugestão, cada correção e principalmente, por ter enxergado (mais do que eu) as maiores qualidades e diferenciais que o trabalho poderia oferecer.

Ao meu colega Danilo, que me acompanhou em grande parte do curso. Cada disciplina cursada e cada trabalho realizado e apresentado foram ótimas experiências ao longo desses anos.

A todos os outros colegas do MCCT, pelo apoio, pelas conversas, pelos aprendizados e pelas experiências vividas.

Aos meus amigos e colegas de trabalho do IFRJ, que sempre me apoiaram e me incentivaram a cursar a pós-graduação. Sou grato por cada experiência que contribuiu para a formação do profissional que me tornei.

Ao meu chefe e amigo de trabalho Leonardo. Faltam-me palavras para dizer o quanto o seu apoio foi essencial para que este trabalho pudesse se concretizar. Muito obrigado por cada sugestão, cada pesquisa, cada teste realizado e cada resultado obtido. Sem a sua ajuda, nada disso teria sido possível.

Ao Denis da DGTI, pelo apoio de sempre e por ter auxiliado na etapa de captura do tráfego da rede. Sua contribuição foi fundamental!

À Sandra, minha terapeuta e amiga. Quando eu mais precisei, você me ajudou a enxergar minhas maiores qualidades, a acreditar em mim mesmo e ir em busca do autoconhecimento. Isso foi fundamental para que eu nunca duvidasse de que eu seria capaz de ter sucesso nessa grande conquista pessoal.

Por fim, agradeço a todos meus familiares, amigos e também a todos que passaram pela minha vida, contribuindo para eu ser a pessoa que sou hoje.

Obrigado!

Henrique Dornel da Silva

Resumo

Lidar com problemas relacionados à segurança da informação de forma corretiva muitas vezes pode implicar na degradação do desempenho e na queda da disponibilidade dos serviços de uma rede de computadores, além de poder gerar altos custos. Portanto, é mais adequado a detecção antecipada e a correção proativa desses problemas, com ganhos na qualidade dos serviços e dos recursos computacionais. O presente trabalho de dissertação apresenta um estudo de previsão de tráfego de dados em uma rede de computadores, utilizando métodos de aproximação conhecidos em análise de séries temporais. Para estimar o tráfego em um determinado momento, foram feitos ajustes e previsões com modelos ARIMA e SARIMA. Mediante a observação de padrões do tráfego da rede, a proposta deste trabalho é desenvolver um sistema computacional capaz de identificar ataques de negação de serviço e comportamentos anormais. Para tanto, as previsões são comparadas com o tráfego real, a fim de detectar possíveis anomalias na rede em um curto período de tempo, quando o tráfego real difere consideravelmente da previsão feita para esse período em particular. A metodologia proposta foi implementada e testada utilizando-se dados reais do tráfego de Internet capturados da borda da rede do IFRJ - Campus Volta Redonda. Os resultados das simulações de ataques comprovaram a viabilidade da utilização da metodologia na prática. Assim, a principal contribuição do sistema desenvolvido é servir como uma importante ferramenta para a segurança da rede. A utilização do sistema na prática auxilia o gerente de rede a favorecer a qualidade das aplicações e serviços disponíveis para os usuários, servindo de suporte na tomada de importantes decisões preventivas de manutenção e melhoria constante da infraestrutura e dos recursos de rede.

Abstract

Dealing with issues related to information security in a corrective way can often cause performance degradation and decrease the availability of services in a computer network, besides possibly generating high costs. Therefore, early detection and proactive correction of these problems are more appropriate, with gains in the quality of services and computational resources. This dissertation work presents a study of prediction of data traffic in a computer network, using known approximation methods in time series analysis. To estimate the traffic at a specific time interval, adjustments and forecasts were made with ARIMA and SARIMA models. By observing network traffic patterns, the purpose of this work is to develop a computational system to identify denial of service attacks and abnormal behaviors. Thereby, the forecasts are compared with the actual traffic in order to detect possible anomalies in the network in a short period of time, when the actual traffic differs considerably from the forecast made for that particular period. The suggested methodology was implemented and tested using real Internet traffic data captured from the network edge of the IFRJ - *Volta Redonda* Campus. The results of the attack simulations proved the feasibility of using the methodology in practice. Thus, the main contribution of the developed system is to serve as an important tool for network security. The use of the system in practice helps the network manager to provide the quality of the applications and services available to users, serving as support in making important preventive maintenance decisions and constant improvement of infrastructure and network resources.

Palavras-chave

1. Redes de Computadores
2. Segurança da Informação
3. Ataques de Negação de Serviço
4. Séries Temporais
5. Modelo ARIMA
6. Modelo SARIMA

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Siglas

ACK	*acknowledgement* (reconhecimento)
AIC	*Akaike Information Criterion* (Critério de Informação de Akaike)
AICc	*Akaike Information Criterion correction* (Critério de Informação de Akaike corrigido)
AP	*Access Point* (Ponto de Acesso)
AR	*Autoregressive* (Autorregressivo)
ARIMA	*Autoregressive Integrated Moving Average* (Autorregressivo Integrado de Médias Móveis)
ARMA	*Autoregressive Moving Average* (Autorregressivo + Médias Móveis)
AT&T	*American Telephone and Telegraph* (Telefonia e Telegrafia da América)
BIC	*Bayesian Information Criterion* (Critério de Informação Bayesiano)
CentOS	*Community Enterprise Operating System* (Sistema Operacional Comunitário Empresarial)
CPU	*Central Processing Unit* (Unidade Central de Processamento)
CSTI	Coordenação de Suporte de Tecnologia da Informação
CUSUM	*Cumulative Sum* (Soma Acumulada)
DDoS	*Distributed Denial of Service* (Negação de Serviço Distribuída)
DIBSeT	Detector de Intrusões Baseado em Séries Temporais
DNS	*Domain Name System* (Sistema de Nomes de Domínio)
DoS	*Denial of Service* (Negação de Serviço)

FAC	função de autocorrelação
FACP	função de autocorrelação parcial
FAPERJ	Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro
FARIMA	*Fractional Autoregressive Integrated Moving Average* (Autorregressivo Integrado de Médias Móveis Fracionário)
FIN	*finalization* (finalização)
GB	Gigabytes
GHz	Gigahertz
HD	*Hard Disk* (Disco Rígido)
HTTP	*Hypertext Transfer Protocol* (Protocolo de Transferência de Hipertexto)
ICMP	*Internet Control Message Protocol* (Protocolo de Mensagens de Controle da Internet)
IFRJ	Instituto Federal de Educação, Ciência e Tecnologia do Rio de Janeiro
INPE	Instituto Nacional de Pesquisas Espaciais
IP	*Internet Protocol* (Protocolo de Internet)
KPSS	Kwiatkowski–Phillips–Schmidt–Shin
MA	*Moving Average* (Médias Móveis)
MAPE	*Mean Absolute Percentage Error* (Erro Percentual Médio Absoluto)
MBps	*Megabits per second* (Megabits por segundo)
MCTIC	Ministério da Ciência, Tecnologia, Inovações e Comunicações
MRTG	*Multi Router Traffic Grapher* (Gerador de Gráficos para Tráfego de Múltiplos Roteadores)
NSFNET	*National Science Foundation Network* (Rede da Fundação Nacional de Ciências)

OCSB	Osborn-Chui-Smith-Birchenhall
PSH	*push* (envio)
QoS	*Quality of Service* (Qualidade de Serviço)
RAM	*Random Access Memory* (Memória de Acesso Aleatório)
Redecomep	Redes Comunitárias de Educação e Pesquisa
RNP	Rede Nacional de Ensino e Pesquisa
RST	*reset* (reinício)
s	segundos
SARIMA	*Seasonal Autoregressive Integrated Moving Average* (Autorregressivo Integrado de Médias Móveis Sazonal)
SDA	Sistema de Detecção de Anomalias
SNMP	*Simple Network Management Protocol* (Protocolo Simples de Gerenciamento de Rede)
SYN	*synchronization* (sincronização)
TCP	*Transmission Control Protocol* (Protocolo de Controle de Transmissão)
TI	Tecnologia da Informação
UDP	*User Datagram Protocol* (Protocolo de Datagrama de Usuário)
UML	*Unified Modeling Language* (Linguagem de Modelagem Unificada)
URG	*urgent* (urgente)
VoIP	*Voice over Internet Protocol* (Protocolo de Voz sobre IP)

Capítulo 1

Introdução

Este capítulo apresenta algumas considerações gerais a respeito do tema abordado neste trabalho, a justificativa para a sua realização e os objetivos a serem alcançados, além de descrever a forma como estão organizados os capítulos seguintes.

1.1 Considerações Gerais

O acesso à Internet nos mais variados ambientes organizacionais tem crescido muito na última década com o avanço tecnológico. Em uma instituição educacional, por exemplo, a Internet e os recursos de rede são fundamentais em diversas atividades acadêmicas e administrativas. O crescimento do uso da Internet e da busca por informação em bases de pesquisa demanda um constante aumento das bandas de conexão, devido à crescente expansão do tráfego de dados na rede, o que impacta diretamente no desempenho dos serviços disponibilizados [1, 2, 3, 4].

A segurança em redes de computadores é um tema muito importante e cada vez mais estudado. Com a expansão da conectividade e troca de informações e com o avanço das tecnologias relacionadas à Internet, o monitoramento do tráfego de dados se torna uma prioridade no gerenciamento de redes e na identificação de anomalias no uso de recursos computacionais [3, 4]. Usuários mal intencionados podem se aproveitar de vulnerabilidades e falhas de segurança na rede ou em sistemas para praticar invasões e ataques, com o intuito de roubar informações ou prejudicar a disponibilidade e o funcionamento dos sistemas [4, 5]. Tais ameaças podem ser identificadas e combatidas através de ferramentas como aplicações *anti-malware* e sistemas de *firewall* e detecção de intrusão. Em contrapartida, um dos grandes desafios atualmente na área é prover mecanismos de segurança eficientes capazes de identificar esses ataques de forma preventiva [2, 6], e proteger os

sistemas computacionais e os usuários de tais ameaças.

Para o reconhecimento de padrões de ataques, o gerenciamento e o monitoramento do tráfego também são de suma importância, pois podem auxiliar o gerente de rede a tomar decisões importantes em casos críticos, como indisponibilidade de recursos e uso inadequado ou incomum da rede [4]. Em suma, há uma grande necessidade de se monitorar as propriedades das redes de computadores a fim de se diagnosticar qualquer problema e gerenciá-los da melhor maneira possível [3].

Uma importante característica do tráfego de rede é a previsibilidade, sendo possível também observar as propriedades de autossimilaridade estatística, dependência de longo período (*long range dependence*) e dependência de curto período (*short range dependence*), conforme citado em [6, 7, 8]. Assim como autores de outros trabalhos encontrados na literatura, eles também afirmam que a previsão de tráfego para um longo período de tempo pode fornecer uma estimativa detalhada de modelos para avaliar os requisitos de capacidade futura e alocação de recursos, servindo como base para decisões estratégicas de planejamento e projeto de redes. Por outro lado, previsões de curto período (milissegundos a minutos) estão relacionadas principalmente à distribuição dinâmica de recursos, controle de congestionamento, mecanismos de *QoS* (*Quality of Service* - Qualidade de Serviço) e detecção de ataques à segurança em redes de computadores, através da comparação do tráfego atual com o tráfego previsto. Esse princípio pode permitir ao gerente de rede tomar providências antes que ocorra degradação de desempenho, congestionamentos ou quedas de conexão [3]. Várias técnicas diferentes são usadas atualmente para análise e previsão de tráfego de rede, incluindo modelos de séries temporais, técnicas de *data mining* (mineração de dados), *machine learning* (aprendizado de máquina) e redes neurais.

A aplicação de técnicas preditivas de controle e monitoramento de tráfego em uma rede de computadores se tornou muito relevante, pois a detecção e correção proativas de problemas são muito mais vantajosas e menos custosas do que o tratamento posterior das falhas e problemas causados pela degradação da rede, conforme afirmado por diversos autores como em [6, 9, 10]. O comportamento do tráfego em uma rede influencia diretamente no desempenho geral dos serviços e recursos. Sendo assim, a análise desse tráfego é uma importante ferramenta para a identificação de eventos maliciosos ou anormais.

Mesmo seguindo padrões recomendados de segurança da informação, muitas instituições ainda carecem de mecanismos mais eficientes e robustos quando o assunto é prevenção a ameaças. Devido ao constante crescimento do número de usuários e do tráfego de dados, novas vulnerabilidades aparecem, fazendo-se necessário o uso de aplicações capazes de de-

tectar ataques e anomalias no tráfego de dados de forma preventiva, antes que ocorram quedas de conexão e indisponibilidade de recursos. É importante também manter a qualidade dos serviços oferecidos, dentro dos limites pré-estabelecidos. Falhas de segurança e eventos maliciosos podem causar a degradação dos serviços e recursos computacionais, além de colocar em risco a integridade dos dados da instituição e dos usuários. Assim sendo, o tratamento posterior e corretivo de falhas e problemas de segurança é muito mais custoso do que se lidar proativamente com anomalias e possíveis ameaças.

Atualmente existem diferentes estudos que mostram que como a coleta de dados trafegados em uma rede é um processo temporal, é possível modelá-los como séries temporais, como os trabalhos [1, 2]. Diversos trabalhos abordam o uso séries temporais como ferramental para detecção de tráfego anormal em redes de computadores, conforme descrito na Seção 2.3. Várias pesquisas mostram que modelos clássicos de previsão por séries temporais como o ARIMA (*Autoregressive Integrated Moving Average* - Modelo Autorregressivo Integrado de Médias Móveis) podem capturar as dependências lineares e de curto período de forma bastante satisfatória [7]. A ideia básica é efetuar previsões de entrada e saída de dados da rede e aplicar os modelos para detecção de anomalias.

1.2 Objetivos

1.2.1 Objetivo Geral

O objetivo principal deste trabalho é propor o desenvolvimento de um sistema computacional para detecção de anomalias e ataques de negação de serviço na rede do Instituto Federal de Educação, Ciência e Tecnologia do Rio de Janeiro (IFRJ) - Campus Volta Redonda, mediante predições por séries temporais.

1.2.2 Objetivos Específicos

Para que o objetivo geral do trabalho possa ser alcançado, os seguintes objetivos específicos devem ser considerados:

- Descrever a metodologia utilizada para detecção de ataques de negação de serviço;
- Realizar a captura em tempo real de dados referentes ao tráfego na rede de computadores da instituição estudada;
- Programar um sistema que realize as seguintes atividades de forma automatizada:
  - Análise e seleção dos dados capturados através de métricas para construção de séries temporais;
  - Realização de previsões de tráfego nas séries e identificação de ataques, através da detecção de anomalias no tráfego real;
  - Envio de e-mails ao gerente de rede sobre a ocorrência de possíveis ataques.
- Validar o método implementado mediante simulações de diferentes ataques em um ambiente real.

1.3 Estrutura da Dissertação

O Capítulo 2 apresenta fundamentos teóricos com definições importantes sobre redes TCP/IP e ataques de negação de serviço, bem como conceitos de séries temporais e modelos de previsão. É apresentada também uma revisão bibliográfica, através da descrição dos principais trabalhos relacionados a esta dissertação.

No Capítulo 3 é descrito o cenário real do estudo de caso e a metodologia proposta neste trabalho é mostrada em detalhes. São descritas as etapas de captura, seleção e análise de dados, construção de séries temporais, detecção de anomalias e identificação de ataques.

As análises e discussões acerca dos resultados obtidos com a metodologia proposta são apresentadas no Capítulo 4, além da validação da metodologia através da simulação de ataques e da apresentação das principais contribuições deste trabalho para o cenário real estudado.

Por fim, o Capítulo 5 apresenta as conclusões finais e propostas de trabalhos futuros.

Capítulo 2

Fundamentos Teóricos e Revisão Bibliográfica

Neste capítulo são apresentados fundamentos teóricos sobre os temas abordados neste trabalho. A primeira parte conceitua redes de computadores, o modelo TCP/IP e seus principais protocolos e apresenta definições sobre tráfego em redes, anomalias e ataques de negação de serviço. A segunda parte aborda séries temporais, modelos de previsão e sua aplicabilidade para o objetivo deste trabalho. Na terceira parte, está presente uma revisão bibliográfica dos principais trabalhos relacionados à dissertação, incluindo aqueles que abordam o uso séries temporais como ferramental para detecção de ataques em redes de computadores.

2.1 Redes TCP/IP

De acordo com [12], uma *rede de computadores* pode ser definida basicamente como "um conjunto de computadores autônomos interconectados por uma única tecnologia". A *Internet* pode ser entendida como um conjunto de redes de computadores interligadas mundialmente, tendo em comum um conjunto de protocolos e serviços [13].

Atualmente, existe uma grande variedade de fabricantes e tipos de dispositivos que estão conectados à Internet, desde supercomputadores, computadores pessoais como *desktops*, *laptops* e *smartphones*, até televisores, veículos e equipamentos domésticos. Os serviços e aplicações oferecidos aos usuários através das redes de computadores e da Internet também são muito diversos, passando por fins de pesquisa e educação, entretenimento e lazer, comércio, compartilhamento de informações e comunicação de modo geral. Nas duas últimas décadas, a Internet vem se tornando cada vez mais importante para a sociedade

e fazendo cada vez mais parte da vida das pessoas.

Devido a essa grande variedade tecnológica, torna-se necessário uma padronização para que a intercomunicação entre os dispositivos conectados à Internet seja possível, através de um modelo de camadas de protocolos. Em uma rede de computadores, define-se *protocolo de rede* como um conjunto de regras para troca de informações, que devem ser seguidas pelos dispositivos interconectados, de forma que possa haver mútuo entendimento das informações trocadas. Segundo [14], em um modelo de camadas, cada camada utiliza os serviços da camada diretamente abaixo dela, que podem ser implementados em *software*, em *hardware*, ou em uma combinação dos dois. O modelo de camadas utilizado na Internet é o *modelo TCP/IP*, e o conjunto de protocolos de todas as camadas formam a *pilha de protocolos TCP/IP*.

2.1.1 Camadas do Modelo TCP/IP

O modelo TCP/IP é composto de cinco camadas: física, enlace, rede, transporte e aplicação [14], conforme mostrado na Figura 2.1.

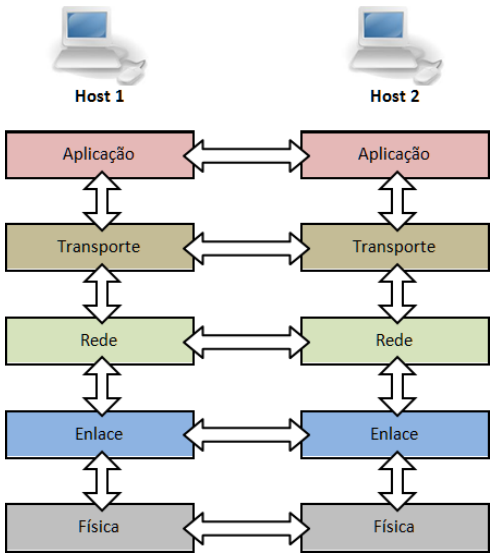


Figura 2.1: Modelo de camadas TCP/IP - Baseado em [14]

- *Camada de Aplicação*: é a camada mais alta do modelo TCP/IP. É composta pelas aplicações de rede que interagem diretamente com os usuários. Exemplos de protocolos que compõe a camada de aplicação são o HTTP (*Hypertext Transfer Protocol* - Protocolo de Transferência de Hipertexto), responsável pela requisição e transferência de documentos pela *web*, e o DNS (*Domain Name System* - Sistema de Nomes

de Domínio), que faz a tradução e gerenciamento de nomes de dispositivos e serviços conectados à rede.

- *Camada de transporte*: é responsável por pegar as informações recebidas pela camada de aplicação, dividir em unidades menores chamadas pacotes, e encaminhar para a camada de rede. Esse procedimento deve ser feito de forma que a camada de aplicação fique isolada das inevitáveis mudanças na tecnologia de *hardware* [12]. Os protocolos da camada de transporte possuem uma visão "fim a fim" de um processo de comunicação [13]. Nessa camada é determinado que tipo de serviço será fornecido: não orientado à conexão ou orientado à conexão. Serviços não orientados à conexão utilizam o protocolo UDP (*User Datagram Protocol* - Protocolo de Datagrama de Usuário) [15]. Serviços orientados à conexão, por sua vez, utilizam o protocolo TCP (*Transmission Control Protocol* - Protocolo de Controle de Transmissão) [15]. Mais detalhes sobre esses dois protocolos serão apresentados na Seção 2.1.2.

- *Camada de rede*: nessa camada é levado em conta qual é a topologia da rede, ou seja, como os dispositivos estão conectados entre si [13]. Os protocolos da camada de rede encaminham os pacotes segundo algoritmos de roteamento e endereçamento. O mais importante deles é o IP (*Internet Protocol* - Protocolo de Internet). O IP é responsável pela interconexão entre as redes, e é único protocolo de endereçamento lógico do modelo TCP/IP. Em [14], os autores afirmam que "a camada de rede é o elemento fundamental que mantém a integridade da Internet". Outro protocolo importante é o ICMP (*Internet Control Message Protocol* - Protocolo de Mensagens de Controle da Internet) [15], utilizado para testar a conectividade entre dispositivos de uma rede, através do comando *ping*, por exemplo. O protocolo ICMP informa se o *host* correspondente ao endereço IP em questão está ou não acessível.

- *Camada de enlace*: a principal tarefa da camada de enlace é transformar um canal de transmissão bruta em uma linha que pareça livre de erros de transmissão não detectados para a camada de rede [12]. Nessa camada também são implementados mecanismos de controle de fluxo e controle de acesso e compartilhamento do meio de transmissão.

- *Camada física*: na camada mais baixa do modelo TCP/IP, os dados recebidos pela camada de enlace são convertidos em sinais digitais, compatíveis com o meio de transmissão utilizado. Em outras palavras, a camada física trata da transmissão de *bits* brutos individuais por um canal de comunicação [12].

2.1.2 Protocolo TCP

Em redes TCP/IP, podem ser oferecidos dois tipos de serviços às aplicações de sistemas finais: serviços não orientados à conexão e serviços orientados à conexão. Em serviços não orientados à conexão, não existe estabelecimento de conexão entre um par de *hosts*, não havendo uma apresentação prévia. Quando um lado de uma aplicação deseja enviar pacotes para o outro lado, o *host* remetente simplesmente os envia [14]. Nesse modelo de comunicação diz-se que a transferência de dados é não confiável. O protocolo de transporte do modelo TCP/IP que fornece um serviço não orientado à conexão é o UDP. Por ser um protocolo mais simples, o UDP é utilizado em aplicações onde os dados precisam ser entregues mais rapidamente, como em *streaming* de vídeo, telefonia VoIP e videoconferência, por exemplo.

Em um serviço orientado à conexão, os *hosts* trocam pacotes de controle entre si antes de iniciar a transmissão dos pacotes com dados reais. No protocolo TCP esse procedimento é chamado de *three-way handshake* (apresentação em três vias), a fim de realizar uma apresentação entre o *host* cliente e *host* servidor. Uma vez concluída a conexão, a comunicação poderá ser efetuada. Esse procedimento será melhor detalhado ao longo desta seção, e está ilustrado na Figura 2.3. Por ser um protocolo orientado à conexão, diz-se também que o TCP proporciona uma transferência de dados confiável, pois a aplicação pode confiar que a conexão entregará todos os seus dados sem erros e na ordem correta [14], através de mecanismos de confirmações e retransmissões. A maioria das aplicações na Internet utiliza o TCP como protocolo de transporte, como por exemplo o HTTP, utilizado na *web*.

Além de conter os dados entregues pelos protocolos da camada de aplicação, os pacotes do protocolo TCP possuem um cabeçalho composto por metadados que são utilizados pelo protocolo para garantir o funcionamento correto dos seus mecanismos. Os metadados do cabeçalho contêm informações como a porta da fonte e a porta de destino para endereçar corretamente os pacotes; número de sequência e número de reconhecimento para garantir a transferência de dados confiável, campos para implementação e controle de fluxo e controle de congestionamento e $flags$ sinalizadoras. Para este trabalho, é importante destacar somente o campo de $flags$, que serão fundamentais na metodologia proposta para detecção de anomalias, descrita em detalhes na Seção 3.3. As $flags$ de um cabeçalho TCP, cada uma correspondente a um *bit*, são mostradas na Figura 2.2 e são detalhadas a seguir.

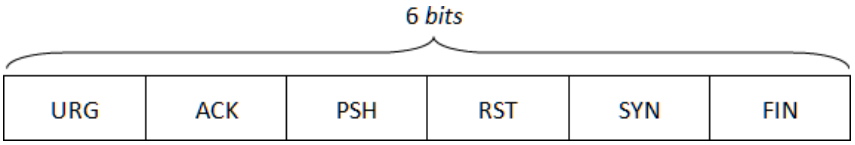


Figura 2.2: *Flags* do cabeçalho do protocolo TCP - Baseado em [16]

- *Flag URG (urgente)*: é usada para identificar dados de entrada como urgentes, de forma que eles não precisem esperar até que os dados anteriores sejam recebidos e sejam enviados diretamente e processados imediatamente [17].

- *Flag ACK (acknowledgement - reconhecimento)*: é usada para confirmar o recebimento bem-sucedido dos pacotes. Através dessa $flag$, é implementado o mecanismo de transferência de dados confiável do TCP. Um exemplo de transmissão de dados confiável entre um *host* 1 e um *host* 2 é descrito a seguir:

  1. Os *hosts* 1 e 2 fazem uma apresentação em três vias e estabelecem uma conexão;
  2. Quando o *host* 2 recebe um pacote do *host* 1, ele envia uma confirmação com um pacote com a $flag$ ACK ativa;
  3. Se o *host* 1 receber a confirmação, ele sabe que o pacote correspondente foi definitivamente recebido;
  4. Se o *host* 1 não receber a confirmação, ele presume que o pacote enviado não foi recebido pelo *host* 2, e faz uma retransmissão.

- *Flag PSH (push - envio)*: essa $flag$ é ativada para solicitar ao host receptor que entregue os dados à aplicação mediante sua chegada imediatamente para a camada superior, em vez de armazená-los até que um buffer completo tenha sido recebido [12]. Assim como a *flag URG*, a *flag PSH* existe para garantir que os dados sejam processados com a prioridade correta. Esse sinalizador é usado com bastante frequência no início e no final de uma transferência de dados [17].

- *Flag RST (reset - reiniciar)*: a $flag$ RST é usada para reinicializar uma conexão incorreta devido a uma falha, para rejeitar um pacote inválido ou para recusar uma tentativa de conexão. Um pacote pode ser rejeitado, por exemplo, quando um *host* recebe um pacote que não era destinado à conexão atual. No protocolo TCP, se for enviado um pacote a um *host* para estabelecer uma conexão e não houver um

serviço aguardando para responder no *host* remoto, este rejeita automaticamente a solicitação e envia uma resposta com o $flag\ RST$ ativada, indicando que a conexão foi redefinida.

- *Flag SYN (synchronization - sincronização)*: é usada para estabelecer conexões entre dois *hosts* no procedimento de *three-way handshake*, ilustrado na Figura 2.3. Inicialmente, o *host* 1 faz uma requisição de conexão enviando um pacote que possui somente a $flag\ SYN$ ativada. A seguir o *host* 2 responde com um pacote com as $flags\ SYN$ e ACK ativadas, confirmando o recebimento da solicitação de conexão. Por fim o *host* 1 confirma o recebimento deste pacote enviando outro pacote, que possui somente a $flag\ ACK$ ativada.

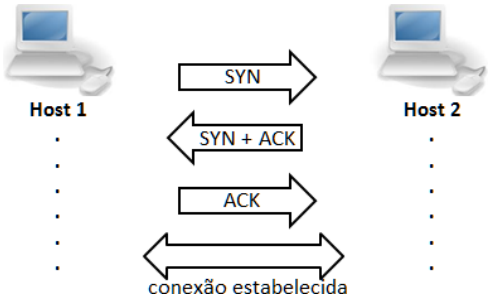


Figura 2.3: *Three-way handshake* do protocolo TCP - Adaptado de [17]

- *Flag FIN (finalização)*: é usada para encerrar uma conexão, indicando que o transmissor não tem mais dados a enviar [12]. A $flag\ FIN$ sempre aparece quando os últimos pacotes são trocados entre *hosts* ou conexões. É importante observar que quando um *host* envia um sinalizador FIN para finalizar uma conexão, ele pode continuar recebendo dados até que o *host* remoto também tenha finalizado a conexão [17]. Uma vez que a conexão é finalizada por ambos os lados, os $buffers$ reservados para a conexão em cada *host* são liberados. A Figura 2.4 descreve um procedimento normal de finalização de conexão: o *host* 1 envia um pacote com as $flags\ FIN$ e ACK ativadas, reconhecendo o fluxo anterior e, ao mesmo tempo, iniciando o procedimento de finalização de conexão, deixando de receber dados a partir deste momento; o *host* 2 responde com uma confirmação somente com a $flag\ ACK$ ativada e em seguida também finaliza a conexão enviando um pacote com as $flags\ FIN$ e ACK ativadas; por fim o *host* 1 responde com outro pacote que possui somente a $flag\ ACK$ ativada.

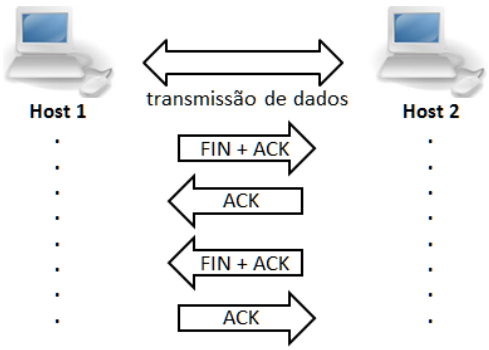


Figura 2.4: Finalização de conexão do protocolo TCP - Adaptado de [17]

2.1.3 Sistemas de Monitoramento de Tráfego

Durante o processo de comunicação em uma rede de computadores, os pacotes são transferidos entre os pares de *hosts*. O conceito de *tráfego* pode ser entendido como grandes conjuntos de dados de pacotes, que são gerados durante a comunicação ao longo do tempo. Dessa forma, a troca de dados entre *hosts* fornece uma sequência finita de pacotes em um determinado intervalo de tempo [2]. Em linhas gerais, o tráfego de rede possui um comportamento dinâmico, no que diz respeito ao número de pacotes trafegados, endereços de origem e destino e protocolos utilizados, por exemplo. Alguns fatores que influenciam o comportamento do tráfego de rede são os tipos de serviços fornecidos, número de usuários e *hosts* conectados e períodos do dia em que os serviços são acessados.

O monitoramento de tráfego consiste na coleta e na investigação dos pacotes transferidos, fornecendo informações adicionais ao gerente de rede através dos resultados analisados, enquanto os serviços de rede são fornecidos continuamente [3]. Existem basicamente dois tipos de sistemas de monitoramento de tráfego de rede:

- *Sistema de monitoramento baseado em SNMP*: em sistemas deste tipo, os dados são coletados através do protocolo SNMP (*Simple Network Management Protocol* - Protocolo Simples de Gerenciamento de Rede) [14]. Este protocolo possibilita a fácil obtenção de informações sobre o tráfego de dados em cada interface de uma rede de forma padronizada e independente de tecnologias proprietárias. Um dos sistemas mais conhecidos é o MRTG (*Multi Router Traffic Grapher* - Gerador de Gráficos para Tráfego de Múltiplos Roteadores) [18], que é uma ferramenta gratuita assim como o Zabbix [19], *software* utilizado atualmente no IFRJ - campus Volta Redonda [11] para este fim. Essas aplicações geram gráficos de forma dinâmica e fornecem informações e estatísticas diversas sobre o tráfego e o estado dos equipamentos da

rede, podendo monitorar o tráfego total ou interfaces específicas do roteador. Um exemplo com gráficos fornecidos pelo Zabbix pode ser viso na Figura 2.5.

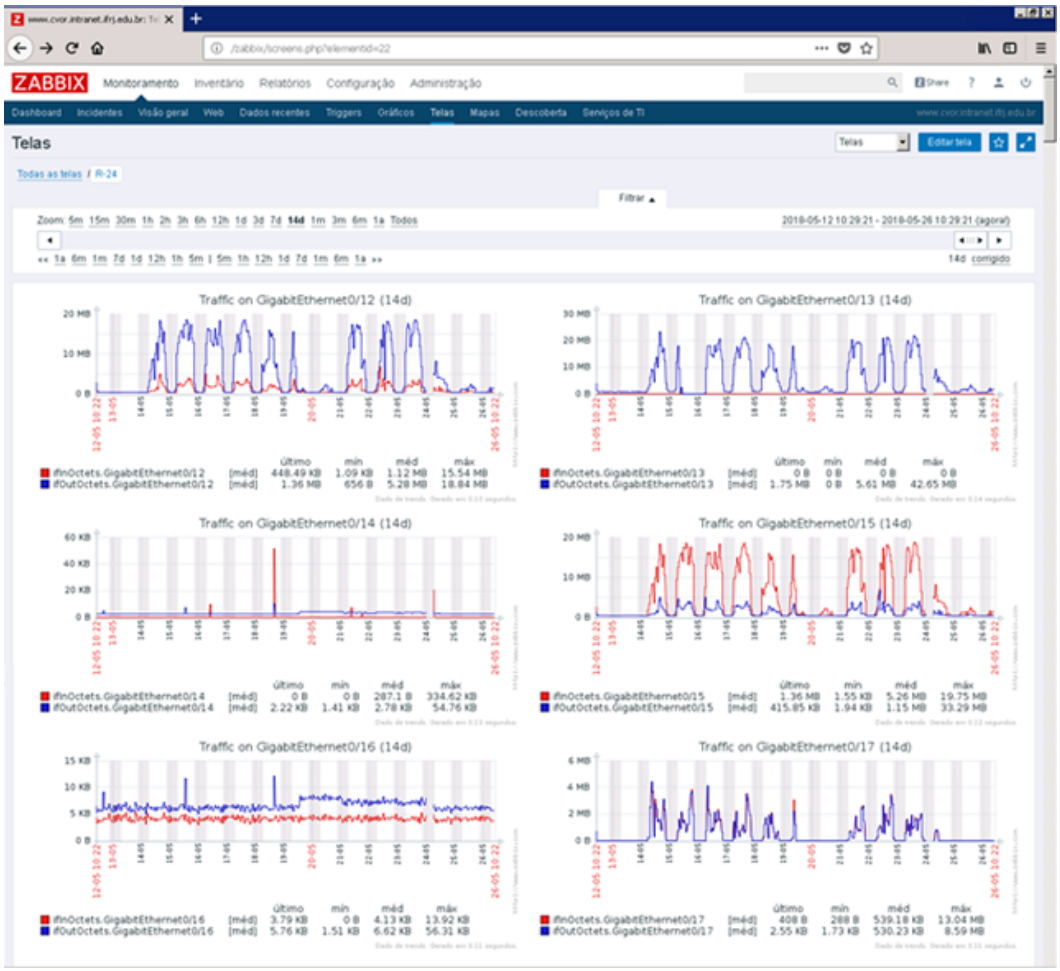


Figura 2.5: Monitoramento de tráfego pelo Zabbix

- *Sistema de monitoramento de pacotes em tempo real*: são sistemas capazes de capturar todos os pacotes trafegados na interface de rede monitorada. A grande vantagem de se utilizar sistemas deste tipo é a possibilidade de captura e análise detalhada de todos os pacotes trafegados, incluindo informações sobre o protocolo e o cabeçalho dos pacotes. A desvantagem é que os *softwares* gratuitos disponíveis não fornecem gráficos detalhados como os principais sistemas baseados em SNMP. Existem diversas opções de *softwares* desse tipo atualmente, incluindo soluções proprietárias de alguns fabricantes, e também soluções *open source*. Nessas ferramentas, a rede pode ser analisada a nível de pacote ou a nível de fluxo [6]. Exemplos conhecidos de *softwares* gratuitos deste tipo são o Wireshark [20] (anteriormente conhecido como Ethereal) e o tcpdump [21]. Um exemplo de captura de pacotes através do Wireshark é mostrado na Figura 2.6.

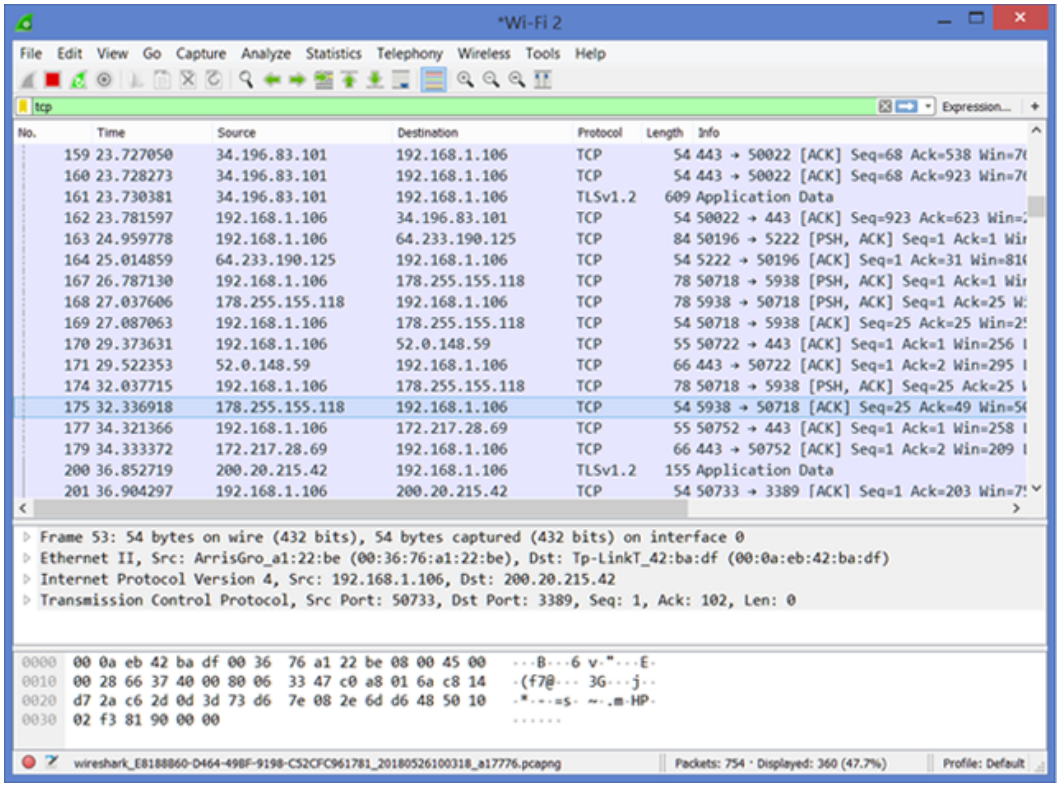


Figura 2.6: Captura de pacotes em tempo real pelo Wireshark

2.1.4 Anomalias no Tráfego

Uma anomalia em um tráfego de rede é um comportamento que difere do comportamento padrão esperado, isto é, que está fora da média prevista para determinada característica da rede [2, 22]. Eventos anômalos podem ser causados por quedas de conexão com a Internet ou com algum servidor, uso inadequado ou incomum dos serviços de rede, falha em recursos de *hardware* ou *software*, infraestrutura física ou mesmo problemas relacionados a medições incorretas no monitoramento do tráfego. Outros eventos que causam anomalias no tráfego são ataques maliciosos feitos por usuários mal-intencionados, como vírus e ataques de negação de serviço. Assim, uma das formas de identificar esses ataques é através da detecção de tráfegos anômalos, de modo que possam ser aplicadas contramedidas para amenizar ou mesmo evitar os danos causados pelos ataques.

A identificação de anomalias no tráfego está diretamente relacionada com a caracterização do tráfego e a maneira de observá-lo [23]. Para diferenciar um tráfego normal de um anômalo, é preciso levar em conta qual característica do tráfego deve ser observada, como por exemplo, a quantidade trafegada de um tipo específico de pacote ao longo do tempo. A Figura 2.7 mostra um exemplo com a quantidade de pacotes UDP trafegados na rede

do IFRJ - Campus Volta Redonda durante duas semanas, com medidas a cada 1 minuto. Esse tráfego é considerado como normal, ou seja, não foram identificadas anomalias nesse período.

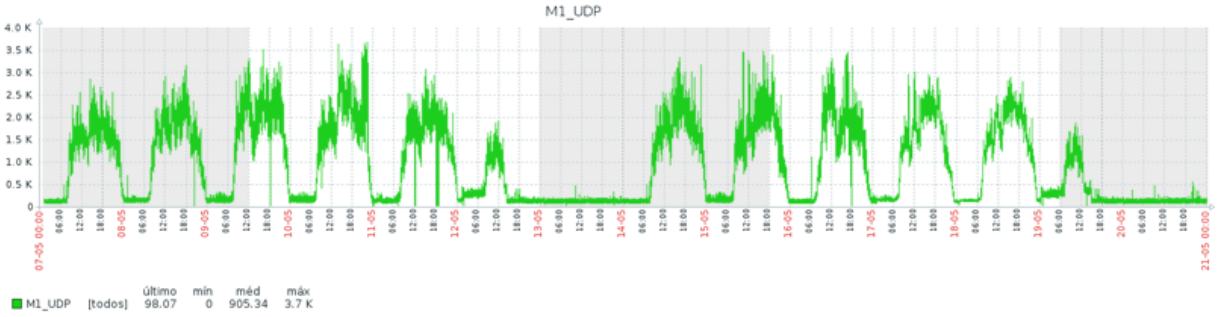


Figura 2.7: Tráfego de pacotes UDP

2.1.5 Ataques de Negação de Serviço

Em redes de computadores, negação de serviço ou DoS (*Denial of Service*) descreve uma situação em que um aplicativo ou servidor de rede não está mais acessível para atender requisições, ou em outras palavras, está negando um serviço [24]. Negações de serviço são geralmente causadas por ataques maliciosos, chamados de *ataque de negação de serviço*, que têm o objetivo de tornar indisponíveis os recursos fornecidos pela vítima [25]. Esses ataques podem ser praticados através da geração de uma sobrecarga no processamento de dados do computador a que o ataque se destina, ou então gerando um grande tráfego de dados que ocupe toda a banda disponível, podendo afetar também outros computadores da mesma rede.

Em um ataque de negação de serviço distribuído (DDoS - *Distributed Denial of Service*), é utilizado um conjunto de computadores atacantes para praticar o ataque, geralmente através de programas maliciosos que realizam sequestro, transformando computadores comuns em zumbis que executam ataques em conjunto, sem a ciência do usuário. Além de aumentar a capacidade do ataque de negação de serviço, essa técnica também reduz a chance de detecção do ataque, já que os pacotes são provenientes de um grande número de computadores pertencentes a usuários insuspeitos [12].

O objetivo dos ataques de negação de serviço não é roubar dados e nem invadir os computadores vítimas, mas sim tornar seus recursos indisponíveis para os outros dispositivos da rede. Um exemplo deste tipo de ataque ocorreu no início de 2000, quando

computadores de várias partes do mundo foram utilizados para indisponibilizar o acesso aos sites de algumas empresas de comércio eletrônico, causando um grande prejuízo [26].

Existe uma variedade muito grande de ataques de negação de serviço que são praticados atualmente. Os mais comuns são baseados em inundação (*flood*) de pacotes. Neste trabalho serão abordados alguns tipos de ataques, descritos a seguir:

- *Fraggle Attack*: é um ataque do tipo *UDP Flood* (inundação UDP), onde um servidor é inundado com um grande número de pacotes UDP falsificados ou fragmentados, com vários endereços IP falsos como origem, com o objetivo de sobrecarregá-lo e deixá-lo indisponível. Como não há estabelecimento de conexão entre os *hosts* no protocolo UDP, é mais difícil para os mecanismos de defesa identificar o ataque. Ataques do tipo *Fraggle Attack* também podem ter como alvo vários servidores de uma rede. Quando um servidor é vítima de um ataque desse tipo, ele não consegue processar todas as solicitações e acaba ficando indisponível após consumir toda sua largura de banda ao enviar vários pacotes ICMP com destino inalcançável.

- *Smurf Attack*: consiste no envio massivo de pacotes *ping* falsificados a partir de um grande número de endereços IP aleatórios como origem. Quando um servidor é inundado por este ataque, seus recursos são esgotados na tentativa de processar as solicitações. Essa sobrecarga reinicia o servidor ou tem um impacto enorme em seu desempenho. Ataques do tipo *Smurf Attack* também podem ter como destino um servidor específico ou vários servidores da rede. Assim como o protocolo UDP, o protocolo ICMP também não possui estabelecimento de conexão, dificultando a detecção do ataque, além do fato de os pacotes enviados se assemelharem facilmente a um tráfego legítimo.

- *DNS Amplification*: é um ataque similar ao *Fraggle Attack*, em que o atacante faz inundações UDP através do envio de uma grande quantidade de pacotes de solicitação DNS falsificados, a partir de um conjunto muito grande de IPs falsos como origem. Como esses pacotes não diferem muito de solicitações legítimas, o *DNS Amplification* é um dos ataques de negação de serviço mais difíceis de detectar e prevenir [24]. Ao tentar atender a todas as solicitações, o servidor esgota seus recursos e fica indisponível.

- *HTTP Floods*: nesse ataque, o atacante utiliza vários endereços IP válidos como origem para estabelecer várias conexões HTTP válidas com um servidor *web*, mantendo a conexão ativa por um longo período de tempo, consumindo e exaurindo os

recursos do servidor. Como os IPs de origem não são falsificados, é mais difícil que os mecanismos de defesa detectem esse tipo de ataque.

- *TCP SYN Flood*: é um dos ataques de negação de serviço mais conhecidos e estudados [25]. Esse ataque explora uma vulnerabilidade do processo de estabelecimento de conexão do protocolo TCP descrito na Seção 2.1.2 e na Figura 2.3, o *three-way handshake*. Para realizar um ataque *TCP SYN Flood*, o atacante inunda a vítima enviando um número muito grande de pacotes TCP com a *flag SYN* ativada, geralmente a partir de endereços IP falsos. Por achar que se trata de solicitações de conexão, a vítima tenta responder a cada solicitação e é ignorada pelo atacante, até esgotar seus recursos e ficar indisponível para processar solicitações legítimas.

- *TCP Reset Attack*: pacotes TCP com a $flag\ RST$ ativa são enviados para encerrar imediatamente uma conexão em caso de algum erro em que seja necessário o encerramento da comunicação entre dois *hosts*. Se o atacante puder visualizar o tráfego da origem para o destino de alguma forma, ele poderá enviar pacotes RST válidos, com o objetivo de encerrar a conexão TCP entre esses *hosts*. Esse é um ataque difícil de ser executado, mas ao fazer isso constantemente, pode-se tornar impossível estabelecer uma conexão.

- *TCP PSH+ACK Flood*: enquanto uma comunicação TCP está ativa, pacotes com as $flags\ PSH$ e ACK ativadas transportam informações entre o *host* de origem e o *host* de destino. Durante um ataque *TCP PSH+ACK Flood*, uma grande quantidade falsificada desses pacotes é enviada para o servidor de destino. Como esses pacotes não estão vinculados a nenhuma conexão válida, o servidor gasta seus recursos ao processá-los, até ficar indisponível para processar pacotes legítimos.

- *TCP FIN+ACK Flood*: para finalizar uma conexão TCP, são enviados pacotes com as $flags\ FIN$ e ACK ativas, conforme mostrado na Seção 2.1.2 e na Figura 2.4. Caso um pacote desse tipo seja enviado sem que uma conexão esteja estabelecida, ele é simplesmente descartado. Mas se um atacante inundar o *host* de destino com um grande número desses pacotes falsificados, a vítima tentará processá-los até esgotar seus recursos. O resultado também é um servidor indisponível para processar solicitações reais.

- *Spoofed Session Attack*: esse tipo de ataque tenta enganar mecanismos de segurança, ao primeiramente estabelecer uma conexão TCP válida. Após o *three-way handshake*, são enviados um ou mais pacotes com as $flags\ RST$ ou FIN ativadas,

mas acompanhados de um número muito grande de pacotes ACK. Esse ataque pode burlar facilmente alguns mecanismos de defesa e esgotar os recursos da vítima, resultando no seu desligamento completo ou em um desempenho inaceitável do sistema.

2.2 Séries Temporais

Uma *série temporal* pode ser definida como um conjunto de observações de uma determinada variável ordenada no tempo, geralmente equidistantes [27], que têm dependência serial, ou seja, dependência entre instantes de tempo. Um *processo estocástico* pode ser classificado como um modelo que descreve a estrutura de probabilidade de uma sequência de observações, isto é, sistemas que evoluem no tempo ou no espaço de acordo com as leis probabilísticas. Pode-se dizer que uma série temporal é o cumprimento de um processo estocástico.

Conforme abordado em [23, 28, 29], uma das maneiras de se classificar as séries temporais é defini-las como contínuas ou discretas. Séries contínuas são utilizadas em abordagens mais teóricas, que fugiriam do foco deste trabalho. Séries discretas podem ser modeladas de duas maneiras:

- Amostragens de dados de natureza contínua. Exemplo: séries pluviométricas, através da modelagem da precipitação de chuva diária em uma determinada região durante o ano;
- Agregação de um conjunto de dados de natureza discreta. Exemplos: preço dos ativos de uma empresa em períodos sucessivos de tempo; e o tráfego de dados em uma rede de computadores, medido em um certo intervalo de tempo ao longo de um período.

Outra forma de classificar séries temporais é como univariadas e multivariadas. Ao modelar uma série como univariada, leva-se em consideração os valores de apenas uma variável aleatória ao longo do tempo. Em séries multivariadas, são considerados os valores de mais de uma variável ou mesmo o valor de outras séries.

Um conceito muito importante no estudo de séries temporais discretas é a granularidade das amostras, que é definida pela periodicidade com que as amostras são obtidas. No caso do tráfego de pacotes em uma rede, a granularidade pode ser definida como segundos, minutos ou horas, por exemplo. Quanto maior for a granularidade, mais detalhes

podem ser analisados sobre o comportamento do tráfego em um período total de tempo determinado.

Para modelar o tráfego de rede como uma série temporal discreta univariada, a variável a ser considerada pode ser o número X de pacotes trafegados em um determinado intervalo de tempo, por exemplo. Como essa variável pode ser considerada como aleatória, infere-se que a série temporal X_t pode ser definida como uma amostragem de um processo estocástico sobre a variável X, isto é, um grupo de variáveis aleatórias X indexadas no tempo t [30]. No entanto, assim como a maioria das séries temporais, o tráfego em redes não é um processo realmente estocástico, uma vez que pode apresentar variações no nível de tendência e sazonalidade.

Uma das maneiras de se decompor uma série temporal é identificando os componentes de tendência, sazonalidade e aleatoriedade. Esses componentes podem se somar ou se multiplicar. É possível então definir um modelo de decomposição aditivo ou um modelo multiplicativo, respectivamente:

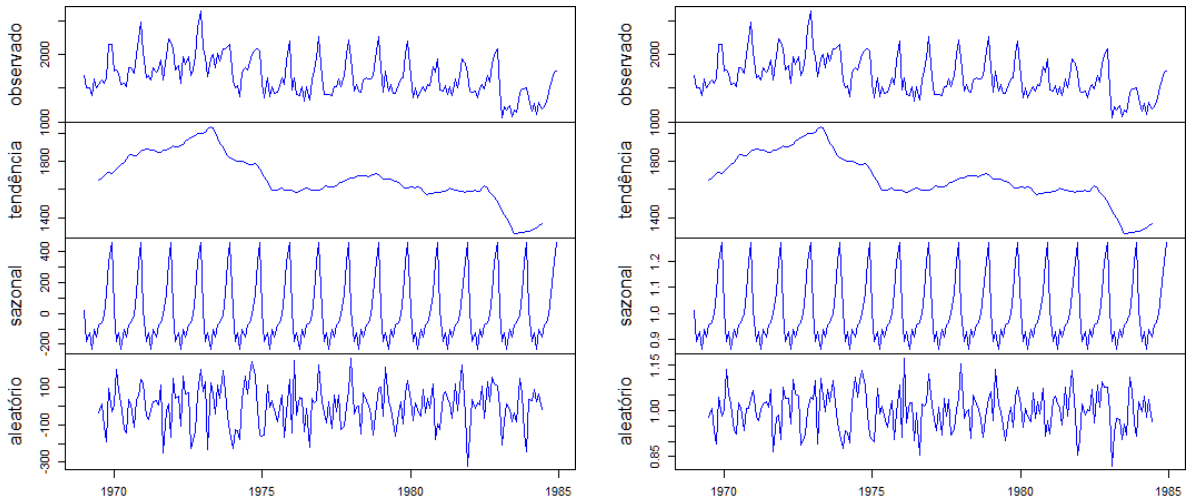
$$\delta_t = T_t + S_t + a_t \tag{2.1}$$

$$\mu_t = T_t \cdot S_t \cdot a_t \tag{2.2}$$

Nas Equações 2.1 e 2.2, δ_t descreve um modelo de decomposição aditivo e μ_t um modelo de decomposição multiplicativo. As componentes de tendência, sazonalidade e aleatoriedade são respectivamente, T_t, S_t e a_t. Modelos aditivos são utilizados quando a variação sazonal é relativamente constante ao longo do tempo, e modelos multiplicativos são mais úteis nos casos em que a variação sazonal aumenta ao longo do tempo.

A Figura 2.8 mostra um exemplo de decomposição de uma série temporal, a saber, a série $UKDriverDeaths$, presente nas bibliotecas da plataforma R [31]. Essa série corresponde ao total mensal de motoristas de automóveis do Reino Unido que foram mortos ou gravemente feridos devido a acidentes de trânsito, no período de janeiro de 1969 a dezembro de 1984 [32].

A componente de tendência pode ser estocástica (estável ao longo do tempo) ou determinística (variável ao longo do tempo), e está relacionada a elementos de longo prazo, podendo ter efeitos constantes, decrescentes ou ascendentes. Pode-se entender a tendência também como a direção da série, ou seja, o rumo que o comportamento da série vai seguir



(a) Decomposição aditiva

(b) Decomposição multiplicativa

Figura 2.8: Vítimas de acidentes de trânsito no Reino Unido: decomposição de série temporal - Adaptado de [32]

no decorrer do tempo. Algumas formas de identificar componentes de tendência são o ajuste de polinômios [29] e o ajuste por reta de mínimos quadrados [33]. A sazonalidade se refere a oscilações ou padrões regulares de curto prazo que se repetem em intervalos iguais de tempo. Efeitos sazonais também podem ser determinísticos ou estocásticos. Todos os outros efeitos randômicos que compõe a série temporal correspondem ao ruído branco, também chamado de componente aleatória, por ser puramente estocástico.

Séries temporais podem ser utilizadas para diversas finalidades, como a obtenção de valores estatísticos; a descrição do comportamento de uma variável de forma gráfica; e a realização de previsões do valor futuro de uma variável ou da série como um todo, através de modelos de previsão.

2.2.1 Processos Estacionários

Nas Equações 2.1 e 2.2, a componente de aleatoriedade dada pela sequência a_t apresenta média zero, variância constante e ausência de correlação serial para cada valor da série temporal [34]. Quando o processo como um todo apresenta essa propriedade, é chamado de estacionário, ou seja, que não possui tendência e nem sazonalidade. Pode-se dizer ainda que um processo é estacionário se todas as características do comportamento do processo não são alteradas no tempo, de modo que o processo se desenvolva aleatoriamente no tempo em torno da média e a escolha de uma origem dos tempos não seja importante, isto é, que suas propriedades não dependam do momento em que foi obser-

vada [33]. Uma série estacionária também é chamada de convergente, em contraste a uma série não estacionária ou divergente. Em uma série não estacionária, a média e a variância são funções do tempo.

Para verificar se uma série possui tendência, podem ser utilizados alguns testes encontrados na literatura, como o Teste KPSS, o Teste de Dickley-Fuller Aumentado e o Teste de Phillips-Perron. Para verificar a presença de sazonalidade, é muito comum a aplicação do Teste OCSB. A apresentação da teoria por trás desses testes foge do escopo deste trabalho, e estão presentes nos devidos referenciais bibliográficos.

Uma das formas de transformar um processo não estacionário em estacionário é aplicando transformações para remover a tendência e a sazonalidade. A maneira mais comum de remover a tendência de uma série temporal é através da aplicação de diferenciações sucessivas para cada valor da série, como mostrado na Equação 2.3, sendo d_t^1 a primeira diferenciação, d_t^2 a segunda diferenciação e assim por diante, até a N-ésima diferenciação d_t^N necessária para remover a tendência da série.

$$\begin{aligned} d_1(X_t) &= X_t - X_{t-1} \\ d_2(X_t) = d(d_1) &= d(X_t - X_{t-1}) = X_t - 2X_{t-1} + X_{t-2} \\ &\dots \\ d_N(X_t) &= d(d_t^{N-1}) \end{aligned} \tag{2.3}$$

Para remover a sazonalidade, pode-se utilizar um processo de diferenciação sazonal, análogo ao procedimento utilizado para remover a tendência. Mas em vez de aplicar a diferenciação com apenas uma defasagem, utiliza-se o período de sazonalidade s. A Equação 2.4 apresenta o procedimento de diferenciação sazonal.

$$\begin{aligned} D_1(X_t) &= X_t - X_{t-s} \\ D_2(X_t) = D(D_1) &= D(X_t - X_{t-s}) = X_t - 2X_{t-s} + X_{t-2s} \\ &\dots \\ D_N(X_t) &= D(D_t^{N-s}) \end{aligned} \tag{2.4}$$

2.2.2 Modelos de Previsão

Realizar previsões em uma série temporal significa fazer estimativas de dados futuros, com base em dados anteriores [29, 35]. Assim sendo, modelos de previsão para séries temporais univariadas baseiam-se no princípio de que padrões históricos de comportamento se repetirão no futuro, para que previsões do valor da variável possam ser efetuadas.

Modelos paramétricos são amplamente utilizados para representar os dados através de uma função matemática, como é o caso da metodologia de Box & Jenkins [36]. A metodologia consiste em realizar ajustes de um conjunto de dados através de um ciclo iterativo para escolha do modelo, com base nos próprios dados [33].

No presente trabalho são utilizados modelos de previsão quantitativos univariados, a partir de dados reais referentes ao tráfego coletado de uma rede de computadores, com base na metodologia de Box & Jenkins, mais especificamente modelos ARIMA e SARIMA.

Modelo AR (Autorregressivo)

Um processo autorregressivo de ordem p, ou $AR(p)$, é dado pela Equação 2.5 [29], onde p é o número de parâmetros do modelo, a_t é o ruído branco e $\phi_1, \phi_2, ..., \phi_p$ são constantes. O termo $\phi(B)$ obtido é chamado de operador autorregressivo.

$$\begin{aligned} Z_t &= \phi_1 Z_{t-1} + \phi_2 Z_{t-2} + ... + \phi_p Z_{t-p} + a_t \\ \phi(B) Z_t &= a_t \end{aligned} \tag{2.5}$$

Este método de aproximação é baseado na premissa de que cada observação em uma série temporal está relacionada a uma ou mais observações anteriores na mesma série. A Equação 2.5 deve satisfazer certas condições para que o processo seja estacionário [36].

Modelo MA (*Moving Average* - Médias Móveis)

Um modelo de médias móveis de ordem q, ou $MA(q)$, é um processo no qual as médias são ajustadas de acordo com as componentes sazonais da série temporal [3]. É um modelo de média móvel ponderada, de número fixo, em que de modo geral, o valor mais recente carrega um peso maior que os valores passados mais distantes. Para uma série temporal estacionária, pode-se usar sua média ou o valor passado imediato como uma previsão para

o período futuro [37].

O processo é descrito pela Equação 2.6. O operador de médias móveis é definido como $\theta(B)$, q é o número de parâmetros do modelo, o ruído branco é dado por a_t e $\theta_1, \theta_2, ..., \theta_q$ são as constantes [29].

$$Z_t = a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - ... - \theta_q a_{t-q}$$
$$Z_t = \theta(B) a_t \tag{2.6}$$

Modelo ARMA (Auto Regressivo + Médias Móveis)

Em uma série temporal estacionária, um modelo $ARMA(p, q)$ requer ambos os termos AR e MA [37], conforme mostrado na Equação 2.7 [29].

$$Z_t = \phi_1 Z_{t-1} + \phi_2 Z_{t-2} + ... + \phi_p Z_{t-p} - a_t - \theta_1 a_{t-1} - \theta_2 a_{t-2} - ... - \theta_q a_{t-q}$$
$$\phi(B) Z_t = \theta(B) a_t \tag{2.7}$$

Modelo ARIMA (Autorregressivo Integrado de Médias Móveis)

Para séries que apresentam comportamento não estacionário devido à componente de tendência, é necessário realizar uma transformação para diferenciá-la e remover a tendência, conforme foi mostrado na Seção 2.2.1. Um modelo indicado para representar a série nesse caso é o $ARIMA(p, d, q)$, onde Δ^d é o operador de diferenciação e o termo d indica o número de diferenciações que a série sofreu para remover a tendência, de acordo com a Equação 2.3. A Equação 2.8 descreve um modelo $ARIMA$ [29].

$$\phi(B) \Delta^d Z_t = \theta(B) a_t \tag{2.8}$$

Modelo SARIMA (Autorregressivo Integrado de Médias Móveis Sazonal)

O modelo $SARIMA(p,d,q)(P,D,Q)_s$ é uma variação do modelo $ARIMA$ que possui também uma parte sazonal, composta por mais três termos. Esse modelo é adequado para séries não estacionárias que possuem comportamento sazonal, com picos e declínios no decorrer do tempo. O período de sazonalidade é dado por s e o termo D corresponde ao número de diferenciações sazonais, calculado pela Equação 2.4.

A Equação 2.9 descreve um modelo $SARIMA$, sendo $\Phi(B^s)$ o operador autorregressivo sazonal, $\Theta(B)$ o operador de médias móveis sazonal e Δ_s^D o operador de diferenciação sazonal [29].

$$\phi(B)\Phi(B^s)\Delta^d\Delta_s^D Z_t = \theta(B)\Theta(B)a_t \tag{2.9}$$

2.2.3 Seleção de Modelos

A análise de séries temporais envolve o estudo de modelos capazes de lidar com uma característica inerente das séries, que é a forte dependência das observações adjacentes, no que se refere aos níveis de autocorrelação. A função de autocorrelação (FAC) é usada para descrever a correlação entre dois valores da mesma série, em função dos valores de defasagem (lag) em que foi calculada. A função de autocorrelação parcial (FACP) tem como objetivo filtrar correlações de outras defasagens, mantendo apenas a correlação pura entre duas observações [33].

Para escolher o modelo ARIMA / SARIMA mais adequado para representar uma série temporal, é necessária a identificação dos termos p, d, q, P, D e Q, de forma que o modelo selecionado para ser aplicado para realização de previsões coerentes, dentro de uma margem de erro aceitável. Conforme explicado anteriormente, os termos d e D correspondem ao número de diferenciações e ao número de diferenciações sazonais, respectivamente. Através dos gráficos das funções FAC e FACP, é possível verificar se os termos autorregressivos e de médias móveis são necessários para corrigir qualquer autocorrelação que permaneça na série, além de identificar suas ordens.

Assim sendo, os números de parâmetros p, q, P e Q do modelo podem ser identificados com base na análise gráfica da função de autocorrelação e da função de autocorrelação parcial da série temporal. A ordem do termo q é estimada a partir dos coeficientes da

função de autocorrelação, podendo ser interpretado como o número de picos significativos nos valores das autocorrelações nas defasagens, dentro do limite de confiança determinado. O termo p é estimado analogamente a partir do gráfico da função de autocorrelação parcial. Os termos P e Q são obtidos de forma análoga aos parâmetros p e q de uma modelo ARIMA não sazonal, mas considerando o número de defasagens como o período de sazonalidade s.

Na metodologia proposta neste trabalho, a seleção de modelos é feita computacionalmente e de forma automatizada, conforme detalhado na Seção 3.3.3. Um exemplo prático de identificação manual dos termos de um modelo é mostrado na Seção 4.2.2, como forma de validar essa parte da metodologia.

Critérios de Seleção

É comum que mais de um modelo ARIMA / SARIMA se ajuste bem à série temporal, sendo necessário identificar qual o modelo que mais se aproxima dos dados reais. Os métodos mais amplamente utilizados para testar a significância dos modelos são o AIC (*Akaike Information Criterion* - Critério de Informação de Akaike) [38] e o BIC (*Bayesian Information Criterion* - Critério de Informação Bayesiano). Os dois métodos têm o objetivo de determinar a probabilidade de se obter os dados de interesse com o modelo analisado. O modelo com melhor ajuste é classificado como aquele em que for calculado o menor valor do critério.

O cálculo do índice AIC considera a qualidade do ajuste aos dados e também o número de parâmetros do modelo, dando preferência para o modelo que tiver menos parâmetros. A qualidade do ajuste é avaliada através da função de máxima verossimilhança, que é definida como a função de densidade de probabilidade conjunta dos dados observados, em função dos parâmetros do modelo. Os estimadores de máxima verossimilhança são definidos como aqueles valores dos parâmetros que são os mais verossímeis, ou prováveis estatisticamente, com relação aos dados observados, maximizando a função de verossimilhança [33]. Mais detalhes sobre o cálculo da função de máxima verossimilhança estão presentes nas referências [9, 39].

A Equação 2.10 mostra como é feito o cálculo desse índice, sendo V a função de máxima verossimilhança para o modelo e k o número total de parâmetros do modelo [40].

$$AIC = -2 \ln V + 2k \tag{2.10}$$

O critério BIC é calculado de forma similar ao AIC, porém com maior penalidade para modelos com maior número de parâmetros. O cálculo do BIC é mostrado na Equação 2.11, onde n é o número de observações da série temporal.

$$BIC = -2 \ln V + k \ln n \tag{2.11}$$

Em [41], o autor propôs uma correção no cálculo do AIC para amostras de número finito, considerando o número de observações no cálculo do critério. A Equação 2.12 apresenta o cálculo do AIC Corrigido (AICc).

$$AICc = -2 \ln V + 2 \frac{k(k+1)}{n-k-1} \tag{2.12}$$

Ao realizar ajustes e previsões, é comum ocorrerem divergências em relação aos valores reais, chamadas de erros ou resíduos. Assim, outra forma de avaliar os ajustes obtidos com um modelo é através do cálculo do MAPE (*Mean Absolute Percentage Error* - Erro Percentual Médio Absoluto), conforme mostrado na Equação 2.13. Considera-se como o modelo mais adequado aquele que apresentar o menor valor MAPE.

$$MAPE = \frac{\sum_{t=1}^{n} \frac{|X_t - \hat{X}_t|}{X_t}}{n} \cdot 100\% \tag{2.13}$$

Na Equação 2.13, X_t é o valor real observado no instante t, $\hat{X}_t$ é o valor ajustado para o instante t e n é o número de amostras da série.

Existem também outros métodos de avaliação de modelos, como a análise de autocorrelação dos resíduos e o Teste de Ljung-Box, que não serão abordados neste trabalho.

2.3 Trabalhos Relacionados

Séries temporais são largamente utilizadas para previsões estatísticas nas mais diversas áreas de pesquisa. Essa aplicabilidade inclui também o problema de previsão de tráfego em redes de computadores, que é abordado por diversos autores desde os anos 1990, como no trabalho [42]. Nesse trabalho já é aplicado o modelo ARIMA, o mesmo abordado nesta dissertação, para previsão em longa escala do número de pacotes trafegados no *backbone* do provedor acadêmico NSFNET.

No trabalho [3], os autores demonstram que modelos ARIMA podem ser usados na previsão do tráfego de rede. Para tanto, eles verificam a satisfação dos modelos pela suposição de estacionariedade das séries, através da avaliação das suas funções de autocorrelação (FAC) e autocorrelação parcial (FACP). O objetivo dos autores é encontrar o modelo mais adequado que possa expressar a natureza do tráfego futuro entre os modelos estudados. Os autores sugerem que o método de previsão proposto pode então ser usado em um protocolo de roteamento como um fator de decisão para o gerenciamento dinâmico de dados de tráfego em uma rede, entretanto eles não chegam a implementar essa proposta no trabalho.

Um trabalho mais recente que utiliza uma abordagem parecida, porém mais completa e atualizada, é o [10]. O autor utiliza o modelo de previsão ARIMA e sua variação SARIMA para apresentar uma metodologia de previsão do tráfego na RedeRio de Computadores/FAPERJ e Redecomep-Rio, além de caracterizar o tráfego observado. Nesse trabalho é destacada a importância de se manter um histórico dos dados coletados, para que se possa realizar com maior precisão previsões para períodos maiores de tempo. O autor obtém resultados através de testes com diferentes escalas de tempo (granularidades) para previsões de períodos de um mês e de seis meses.

Outras técnicas são apresentadas em outros estudos, como por exemplo em [7], onde os autores fazem um estudo baseado na comparação de diversas abordagens de previsão para o tráfego de Internet. Nesse trabalho são construídos diferentes modelos de previsão baseados em FARIMA (ARIMA Fracionário) e redes neurais artificiais. A primeira abordagem utiliza técnicas para comparação dos dois modelos, sendo que uma segunda abordagem refere-se a modelos híbridos baseados na combinação desses modelos. Além disso, as abordagens são comparadas com outras alternativas bem conhecidas para previsão do tráfego por séries temporais, como ARIMA e Holt-Winters. Os autores mostram que os modelos híbridos construídos são mais adequados quando se observa a característica

de não-linearidade nas séries. Essa característica não foi observada nos dados capturados para realização da metodologia proposta no presente trabalho de dissertação.

Em [43], é feito um estudo de previsões de curto prazo do tráfego em redes de computadores, através da comparação de resultados obtidos com redes neurais e modelos lineares de séries temporais baseados em ARIMA e suavização exponencial. Através de dados e resultados experimentais, a autora mostra que, na maioria dos casos, as diferenças entre a qualidade dessas previsões produzidas por redes neurais e pelos modelos lineares não são estatisticamente significativas. Ela conclui ainda que a aplicação de métodos mais complexos e computacionalmente custosos, como redes neurais, pode não ser a mais apropriada para previsões de curto prazo em tráfego de redes.

No trabalho [2], são apresentados resultados de uma pesquisa realizada no INPE (Instituto Nacional de Pesquisas Espaciais) que visa caracterizar o padrão de comportamento do tráfego de uma rede. Para tanto, o tráfego é agrupado em quatro períodos do dia, sendo cada um representado por um conjunto de nove atributos. Utilizando técnicas de análise de séries temporais e inteligência computacional, são realizados os cálculos das faixas de valores consideradas como padrão para cada atributo em cada período.

Em [1], os autores ressaltam a necessidade de se monitorar o tráfego de uma rede de computadores em tempo real, mas destacam a dificuldade de implementação em redes com maior tráfego. Os autores propõem uma maneira de superar esse problema, coletando amostras de dados referentes ao tráfego de forma seletiva. Nesse trabalho, é mostrado que é possível representar com precisão o tráfego de uma rede de alta velocidade usando modelos adequados de séries temporais e, em seguida, determinar o tamanho da janela de amostragem, com o objetivo de detectar perda de pacotes em tempo real.

Os autores do trabalho [6] apresentam uma revisão de várias técnicas propostas, utilizadas e praticadas para análise e previsão de tráfego de rede em outros trabalhos, incluindo análise de séries temporais, técnicas baseadas em redes neurais e modelos híbridos.

Também estão presentes na literatura vários trabalhos relacionados mais diretamente à proposta apresentada nesta dissertação. A aplicação de séries temporais para detecção de anomalias em redes de computadores tem sido cada vez mais abordada, desde os anos 2000.

Os primeiros trabalhos que motivaram o estudo do tema aqui abordado foram [44, 45]. O autor desses trabalhos faz um estudo da identificação de possíveis ocorrências dos ataques de negação de serviço *TCP SYN Flood*, *Smurf Attack* e *Fraggle Attack*. Essa

identificação se dá a partir da detecção de anomalias em séries temporais referentes ao tráfego de pacotes em uma rede. A utilização de séries temporais foi escolhida por ser pouco custosa a nível computacional. A ideia é estabelecer limites estáticos dinâmicos ao comportamento futuro do tráfego, e basear-se no princípio de que quando um número de pacotes trafegados ultrapassa um limite superior estabelecido, em relação à curva criada através de análises passadas de normalidade, é um indicativo de uma possível anomalia causada por um ataque. O modelo de previsão utilizado é o ARIMA e foi desenvolvido um *software* na linguagem Java, batizado pelo autor de DIBSeT - Detector de Intrusões Baseado em Séries Temporais. Apesar de obter bons resultados, o autor teve dificuldades em relação ao processo de captura dos dados referentes ao tráfego.

No trabalho [46], o objetivo também é detectar anomalias em redes de computadores. Modelando o tráfego como séries temporais, é proposta uma metodologia que consiste na previsão de valores futuros em um curto período de tempo, utilizando a técnica de Holt-Winters. Um tráfego é classificado como anormal quando os valores reais medidos se desviam muito dos valores previstos.

Os autores de [25] fazem um estudo de ataques de negação de serviço, mas especificamente o *TCP SYN Flood*. Nesse artigo, os autores também mostram que a detecção de ataques desse tipo pode ser feita empregando técnicas de limite adaptativo e algoritmo $CUSUM$. A utilização dessas técnicas, entretanto, necessita da definição de um limite para os pacotes TCP SYN recebidos, e carecem de melhorias e desenvolvimento de métodos mais robustos.

Uma das principais obras relacionadas ao presente trabalho é a [23], em que o autor propõe o desenvolvimento de um sistema que se baseia em anomalias no tráfego para detecção de ataques de negação de serviço e escaneamento de porta. O método proposto pelo autor é divido em quatro etapas: coleta, classificação, obtenção de métricas e análise. Na etapa de coleta, o autor realiza a captura dos fluxos referente ao tráfego da borda da Rede-Rio de Computadores. Em uma rede TCP/IP, um fluxo de pacotes pode ser entendido como o conjunto de pacotes enviados numa transmissão desde sua abertura até seu encerramento, segundo definição do próprio autor. Diferentemente dessa abordagem, no presente trabalho a captura do tráfego não é feita a nível de fluxos, mas sim a nível de pacotes, conforme detalhado na Seção 3.3.1.

Ainda em [23], com o objetivo de isolar eventos anômalos e facilitar a detecção de anomalias, o autor faz um agrupamento dos fluxos baseado no trabalho [47], em que a autora propõe uma análise de tráfego em redes para classificação dos fluxos em quatro

partições. E para definição de quatro métricas, o autor recorre ao trabalho [48]. Dessa forma, são construídas dezesseis séries temporais, sendo quatro séries referentes a essas métricas para cada uma das quatro partições definidas na etapa de classificação. A seguir, as séries temporais são analisadas e são feitas previsões com o modelo de Holt-Winters. A cada 5 minutos, o comportamento do tráfego é então classificado como comum ou anômalo, a fim de identificar diferentes ataques, sendo que alguns deles também são abordados aqui neste trabalho.

No artigo [4] também são tratados dois tipos de ameaças: ataques de negação de serviço e escaneamento de porta. É proposto um método para detecção de anomalias baseado na análise do tráfego a nível de pacotes e a nível de fluxos, e é desenvolvido um sistema que analisa a variação da entropia de informações referentes ao tráfego capturado. Para tanto, os autores propõem quatro métricas relacionadas à identificação de eventos maliciosos.

O trabalho desenvolvido nesta dissertação tem foco em previsões de curto prazo para o tráfego de pacotes em uma rede. Utilizando modelagem por séries temporais com modelos ARIMA e SARIMA, o objetivo principal é a detecção em tempo real de diferentes ataques de negação de serviço, a cada intervalo de tempo de 1 minuto. A metodologia proposta para alcançar esse objetivo é descrita em detalhes no próximo capítulo.

Capítulo 3

Metodologia Proposta

A metodologia proposta nesta dissertação é inspirada principalmente nos trabalhos relacionados descritos na Seção 2.3, em especial aqueles em que os autores aplicam séries temporais para modelagem do tráfego em redes de computadores. No presente trabalho é proposta a utilização de métricas próprias, com foco na identificação de ataques de negação de serviços específicos. O trabalho consiste no desenvolvimento do protótipo de um sistema automatizado, capaz de identificar ataques com base em tráfego anormal de pacotes. O sistema foi batizado de SDA (Sistema de Detecção de Anomalias).

O cenário de estudo, a metodologia e as etapas de desenvolvimento são detalhados neste capítulo. A primeira parte do capítulo lista os princípios básicos que regem o desenvolvimento do sistema. A segunda parte descreve o cenário real do ambiente onde se propõe a aplicação do método, detalhando o panorama atual do monitoramento da rede do IFRJ - Campus Volta Redonda. A terceira parte aborda em detalhes os módulos do sistema desenvolvido, descrevendo as etapas de captura do tráfego, seleção de dados, definição de métricas, construção de séries temporais e detecção de anomalias e ataques.

3.1 Princípios Básicos

Para o desenvolvimento do Sistema de Detecção de Anomalias, este trabalho segue os seguintes princípios básicos:

- Utilização de *softwares* e linguagens de programação livres;
- Independência de tecnologias proprietárias;
- Baixo custo operacional;

- Simplicidade de implementação;
- Facilidade de manutenção;
- Alto desempenho computacional;
- Escalabilidade.

3.2 Cenário de Estudo

Na rede de computadores do campus Volta Redonda do IFRJ, assim como em qualquer instituição de ensino típica, são constantemente utilizados diversos serviços e aplicações baseados na Internet, disponíveis para estudantes, funcionários e comunidade externa, equivalentes a aproximadamente 400 usuários simultâneos nos horários de maior utilização. As conexões podem ser cabeadas ou sem fio, em computadores institucionais ou pessoais, dispositivos móveis e aparelhos telefônicos VoIP. A infraestrutura de rede inclui um total de 13 *switches* (sendo 10 gerenciáveis), 9 pontos de acesso (*access points*) para conexão *wireless*, um roteador central, um *firewall* e diversos servidores. Um esquema da infraestrutura da rede do campus é mostrado na Figura 3.1.

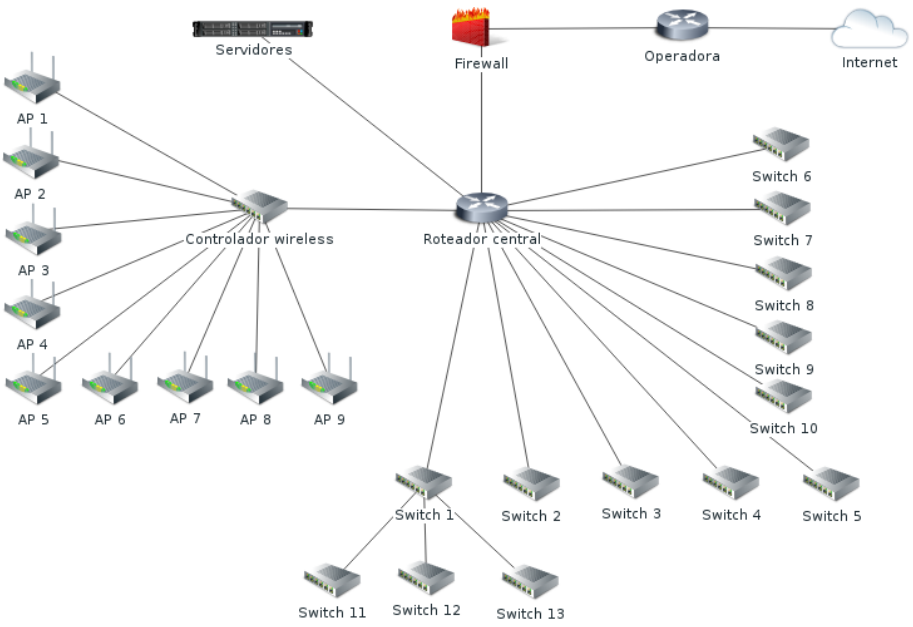


Figura 3.1: Infraestrutura de rede do IFRJ - Campus Volta Redonda

O *link* de Internet é dedicado e *full duplex*, possui capacidade de transmissão de 20MBps e é fornecido pela Rede Nacional de Ensino e Pesquisa (RNP) [49], através de uma empresa terceirizada. A RNP é uma organização social vinculada ao Ministério da Ciência, Tecnologia, Inovações e Comunicações (MCTIC), responsável pelo *backbone* da

rede acadêmica brasileira, que inclui instituições públicas de pesquisa e de ensino superior e tecnológico.

Conforme explicado na Seção 2.1.3, monitorar o tráfego em uma rede consiste em coletar e investigar os pacotes transferidos. Com o objetivo de monitorar o tráfego no campus, a Coordenação de Suporte de Tecnologia da Informação (CSTI) utiliza o Zabbix, que é um *software* livre para monitoramento baseado em SNMP. A utilização do Zabbix tem como objetivo monitorar diversos recursos da rede, como o desempenho e o estado de servidores e equipamentos. São apresentados gráficos com informações bem diversas, como por exemplo, a taxa de *byte*s por segundo dos dados trafegados na borda da rede ou em uma interface específica, em determinados intervalos de tempo. Na Figura 2.5 foram mostrados exemplos de gráficos gerados pelo Zabbix.

A instituição já segue parâmetros rígidos de segurança em redes, mas ainda é necessária a implantação de um sistema capaz de automatizar a detecção de ameaças, com o objetivo de servir de suporte para o gerente de rede e auxiliá-lo na tomada de decisões. Assim sendo, com a realização do presente trabalho, pretende-se buscar uma forma de evitar ou amenizar certos problemas relacionados a ameaças à segurança da informação, fornecendo uma ferramenta para auxiliar na detecção e prevenção de alguns casos comuns de ataques de negação de serviço, mais especificamente na instituição de ensino estudada.

3.3 Sistema de Detecção de Anomalias

Essa seção descreve em detalhes os três módulos que compõe o SDA: captura de tráfego, seleção de dados e detecção de anomalias. Todos os módulos do sistema operam de forma simultânea, programada e automatizada, de forma que não seja necessária a intervenção humana em nenhum dos processos.

Toda a aplicação é executada a partir de um computador destinado somente para este trabalho, localizado na CSTI do campus Volta Redonda. Chamaremos esse computador de servidor SDA. Suas configurações são as seguintes: sistema operacional Linux CentOS 7 [50], processador com quatro núcleos de 3,7GHz, 4GB de memória RAM, HD de 500GB e placa de rede *Gigabit Ethernet*.

A Figura 3.2 contém um fluxograma em formato de Diagrama de Atividades UML (*Unified Modeling Language* - Linguagem de Modelagem Unificada) [55], que resume todas as etapas do funcionamento do SDA, proposto nesta seção.

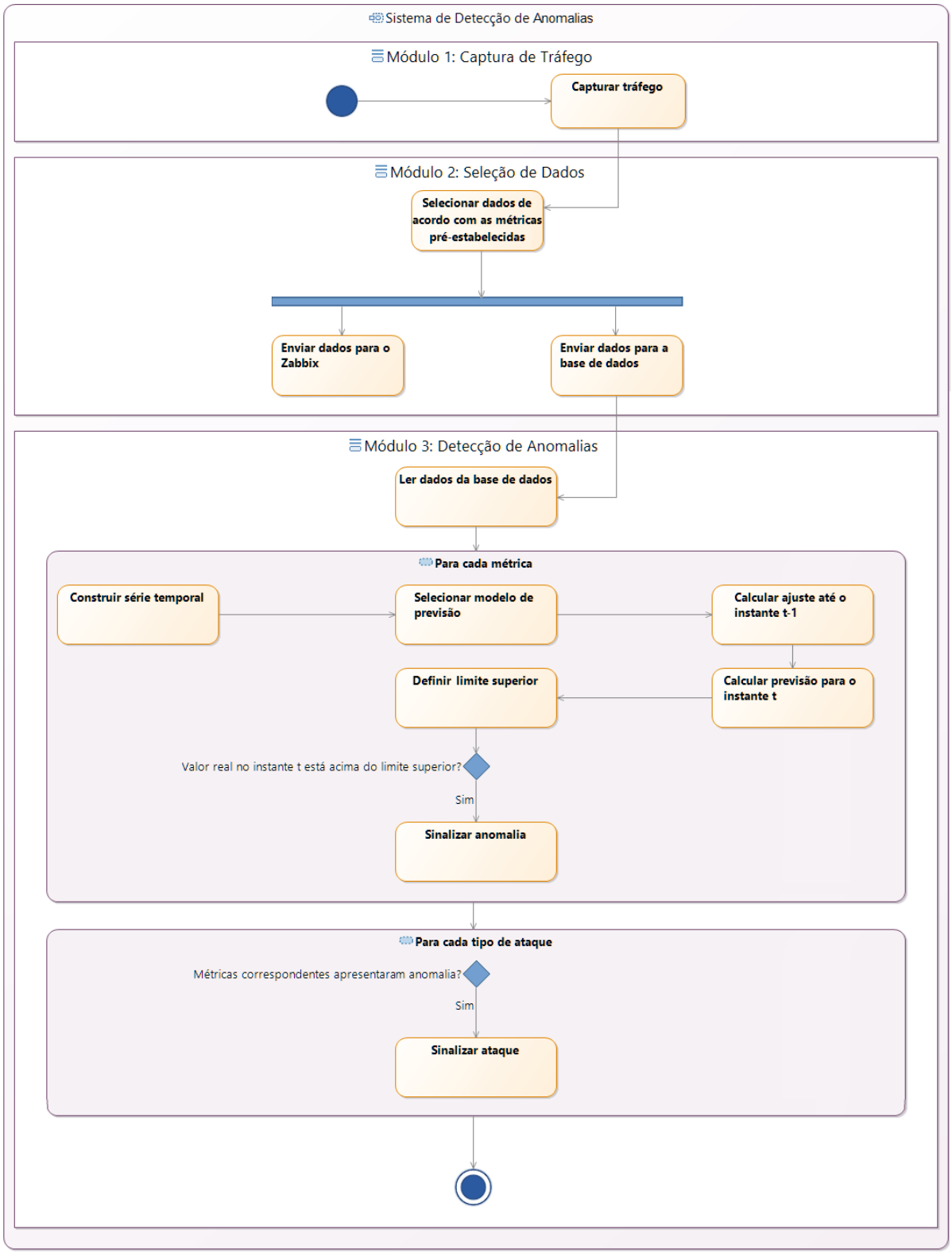


Figura 3.2: Diagrama de Atividades UML, com um resumo de todas as etapas da metodologia proposta

3.3.1 Módulo 1: Captura de Tráfego

Conforme definido na Seção 2.1.3, a comunicação entre *hosts* de uma rede fornece um conjunto finito de pacotes em um determinado intervalo de tempo. O primeiro desafio encontrado no desenvolvimento deste trabalho foi a captura em tempo real dos dados referentes ao tráfego na borda da rede do campus Volta Redonda do IFRJ. Conforme foi explicado na Seção 3.2, atualmente no campus é utilizado o Zabbix para monitoramento do tráfego. O problema é que este *software* não faz a captura dos pacotes trafegados em tempo real de forma nativa. A única informação fornecida sobre o tráfego em si é a taxa média de *bytes* por segundo transmitidos, dentro de um intervalo de tempo especificado. Essa informação não é suficiente para o objetivo principal desse trabalho, que é identificar ataques de negação de serviço baseado em anomalias no tráfego. Lidar somente com séries temporais baseadas no total de dados trafegados permitiria somente fazer a detecção de anomalias muito genéricas e poderia gerar um número muito grande de falsos positivos e falsos negativos, tornando o método ineficaz. Além disso, não seria possível identificar tipos de ataques específicos somente com esses dados.

Por esse motivo, foi necessária a utilização de um sistema de monitoramento de pacotes em tempo real, capaz de capturar o tráfego de forma mais detalhada, tanto de entrada quanto de saída. O *software* escolhido para este trabalho foi o TShark [20], dada a sua facilidade de integração com o Zabbix. O TShark captura o tráfego a nível de pacotes, e é uma versão do Wireshark baseada em linha de comando, não possuindo interface gráfica. Foram descritos mais detalhes sobre os tipos de sistemas de monitoramento de tráfego na Seção 2.1.3.

O TShark foi instalado no servidor SDA, sendo responsável por capturar todo o tráfego que passa pela borda da rede. Para isso, o servidor foi conectado a uma interface espelhada da interface correspondente ao link de Internet do roteador central, conforme mostrado na Figura 3.3. Como são capturados dados referentes ao tráfego da borda da rede, o método proposto não será eficaz na identificação de ataques com origem e destino dentro da própria rede.

O primeiro módulo do SDA, referente à captura do tráfego, foi escrito na linguagem *Shell Script* do Linux. O *script* basicamente faz a captura de todo o tráfego da rede, ou seja, todos os pacotes que entram e saem da rede são capturados. As informações sobre os pacotes são salvas em arquivos no formato do TShark, que são armazenados temporariamente no HD do servidor a cada intervalo de tempo de 1 minuto.

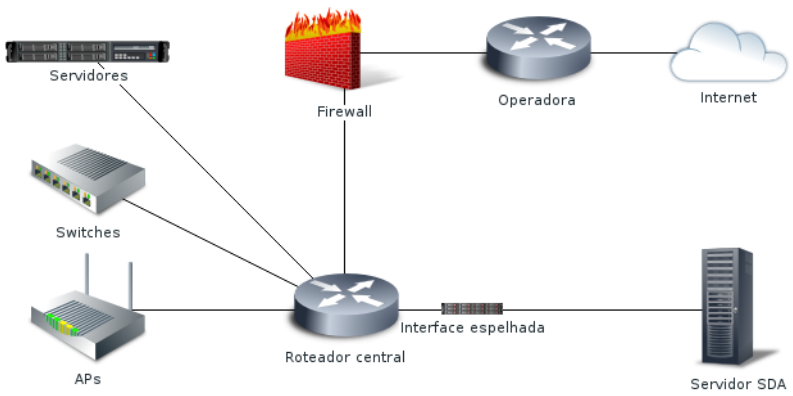


Figura 3.3: Conexão do servidor SDA ao roteador central

É importante destacar que os dados capturados correspondem somente aos metadados dos pacotes que trafegam pela rede. De forma alguma o conteúdo dos pacotes é monitorado, visualizado e nem disponibilizado, mantendo-se assim a privacidade dos usuários da rede. Os outros módulos do sistema serão detalhados no decorrer deste capítulo.

3.3.2 Módulo 2: Seleção de Dados

Para que as anomalias que ocorram no tráfego possam ser identificadas, de forma a possibilitar a detecção de ataques específicos, é necessário selecionar quais dados referentes ao tráfego serão analisados. Os arquivos gerados pelo TShark são referentes a todo o tráfego da rede. Os pacotes são enumerados e são coletadas todas as informações sobre eles, inclusive os metadados contidos nos cabeçalhos referentes aos protocolos de todas as camadas do modelo TCP/IP. Entretanto, somente algumas informações são necessárias para a execução da próxima etapa do método.

O segundo módulo do sistema, também escrito em *Shell Script*, é responsável pela extração e seleção dessas informações, que serão analisadas no módulo posterior. Através da leitura em tempo real de cada arquivo gerado pelo módulo de captura de tráfego, são obtidas treze informações de interesse a cada 1 minuto. Cada uma dessas informações corresponde a uma métrica.

Nessa etapa a seleção é feita com base no princípio de que cada um dos tipos de ataque de negação de serviço abordados nesse trabalho pode ser identificado a partir de uma ou mais métricas correspondentes, sendo que cada uma das métricas pode referir-se à identificação de um ou mais ataques específicos. Ao lidar com informações mais gerais sobre o tráfego, o impacto de um ataque ficaria muito mais diluído, dificultando a sua detecção. Por esse motivo, a definição de métricas tem o objetivo de aumentar a

capacidade de identificação de ataques.

Cada métrica é obtida através da contagem do número total de pacotes recebidos e enviados (entrada e saída) em um intervalo de tempo de 1 minuto, de acordo com as regras definidas para cada uma delas. Neste trabalho, é proposta a definição de onze métricas principais, que são listadas a seguir:

- *UDP*: total de pacotes transmitidos que utilizam o UDP como protocolo na camada de transporte.
- *ICMP*: total de pacotes transmitidos que utilizam o ICMP como protocolo na camada de rede.
- *DNS*: total de pacotes transmitidos que utilizam o DNS como protocolo na camada de aplicação.
- *HTTP*: total de pacotes transmitidos que utilizam o HTTP como protocolo na camada de aplicação.
- *TCP_ACK*: total de pacotes transmitidos que utilizam o TCP como protocolo na camada de transporte e possuem somente a $flag\ ACK$ ativada.
- *TCP_SYN*: total de pacotes transmitidos que utilizam o TCP como protocolo na camada de transporte e possuem somente a $flag\ SYN$ ativada.
- *TCP_SYN_ACK*: total de pacotes transmitidos que utilizam o TCP como protocolo na camada de transporte e possuem as $flags\ SYN$ e ACK ativadas.
- *TCP_RST*: total de pacotes transmitidos que utilizam o TCP como protocolo na camada de transporte e possuem somente a $flag\ RST$ ativada.
- *TCP_RST_ACK*: total de pacotes transmitidos que utilizam o TCP como protocolo na camada de transporte e possuem as $flags\ RST$ e ACK ativadas.
- *TCP_PSH_ACK*: total de pacotes transmitidos que utilizam o TCP como protocolo na camada de transporte e possuem as $flags\ PSH$ e ACK ativadas.
- *TCP_FIN_ACK*: total de pacotes transmitidos que utilizam o TCP como protocolo na camada de transporte e possuem as $flags\ FIN$ e ACK ativadas.

Conceitos sobre os protocolos de rede utilizados para obtenção das métricas principais foram mostrados em mais detalhes na Seção 2.1.

Além dessas métricas, também foram definidas duas métricas secundárias, que servirão para análise das métricas principais na Seção 4.1:

- *OUTROS_PACOTES*: total de todos os outros pacotes transmitidos.
- *TOTAL_PACOTES*: total de todos os pacotes transmitidos, representando o tráfego total de pacotes.

A seguir o *script* envia os valores referentes a cada uma dessas métricas pré-estabelecidas para o Zabbix, através de uma integração a partir de *template* [19]. Aqui o objetivo da utilização do Zabbix é somente a visualização dos dados através de gráficos, conforme mostrado na Figura 3.4. Os mesmos dados também são armazenados em uma base de dados local MySQL/MariaDB [51], para então construir-se as séries temporais referentes a cada métrica, no módulo seguinte.

É importante destacar que os gráficos da Figura 3.4 representam dados referentes a um tráfego de rede considerado como normal para as métricas analisadas, durante um período de duas semanas. No período de exemplo observado, nenhum uso da rede foi classificado como anormal e nem ocorreram quedas significativas de conexão. Além disso, é possível observar períodos de repetição do comportamento do tráfego, devido à baixa utilização da rede aos finais de semana e durante a madrugada, e uso mais intenso principalmente em dias de semana entre 9h e 20h.

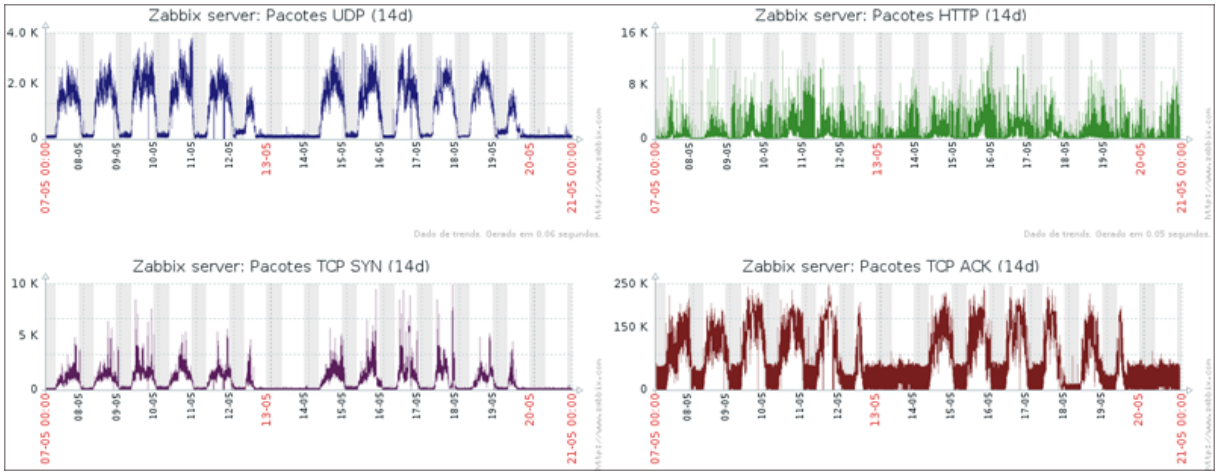


Figura 3.4: Gráficos gerados pelo Zabbix, referente ao tráfego de pacotes das métricas *UDP*, *HTTP*, *TCP_SYN* e *TCP_ACK*

3.3.3 Módulo 3: Detecção de Anomalias

O terceiro e último módulo do SDA é responsável pela detecção de anomalias nas séries temporais e pela identificação de um possível ataque correspondente às anomalias. Esse módulo do sistema foi desenvolvido com a linguagem de programação R [31].

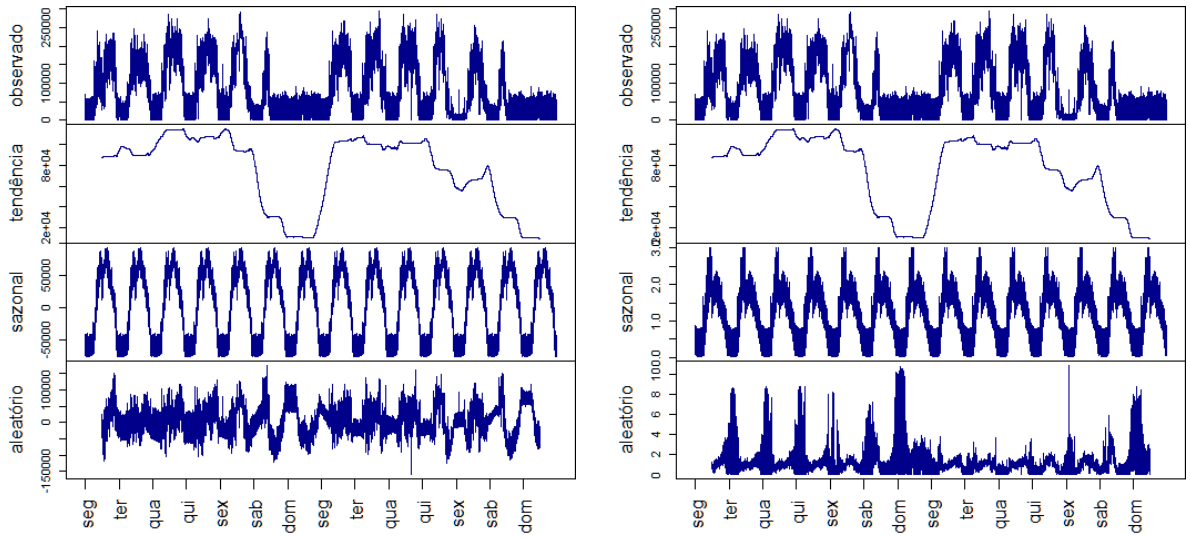
Além de ser uma linguagem de programação livre, o R é uma plataforma de desenvolvimento integrado para computação estatística, disponível para diferentes sistemas operacionais. É um projeto GNU similar à linguagem S, que foi desenvolvida na Bell Laboratories (antiga AT&T, atual Lucent Technologies). A linguagem S é frequentemente o veículo de escolha para pesquisa em metodologia estatística, sendo que o R fornece uma poderosa solução *open source* para esta finalidade. O R possui nativamente uma ampla variedade de técnicas estatísticas e gráficas, como modelagem linear e não linear, testes estatísticos clássicos e análise de séries temporais, além de ser altamente extensível. Outro ponto forte do R é a facilidade e a qualidade da plotagem de gráficos, sendo ideais para utilização em publicações científicas [31].

Construção de Séries Temporais

Nesta etapa, a construção das séries temporais é feita utilizando-se a função *zoo* do R [31, 32]. Essa função basicamente armazena os dados em dois componentes: um vetor de intervalos de tempo e um conjunto de dados referentes aos valores observados das amostras.

Cada uma das treze métricas pré-definidas, cujos valores foram obtidos no módulo anterior, tem seu valor indexado em relação ao intervalo de tempo em que foi obtida, para a construção de uma série temporal correspondente. A granularidade escolhida para construção das séries temporais foi de 1 minuto, o mesmo intervalo de tempo em que é obtida cada métrica, com o objetivo de minimizar o tempo de identificação da ocorrência de um ataque.

O *R* possui uma função nativa chamada *decompose*, que faz a decomposição de uma série temporal em componentes de tendência, sazonalidade e aleatoriedade. A presença dessas componentes pôde ser observada nas séries temporais de todas as métricas, evidenciando a característica de não-estacionariedade. A Figura 3.5 contém a decomposição da série temporal da métrica $TOTAL\_PACOTES$, referente a um período de duas semanas.



(a) Decomposição aditiva

(b) Decomposição multiplicativa

Figura 3.5: Decomposição da série temporal da métrica $TOTAL\_PACOTES$

Previsões com Modelos ARIMA / SARIMA

Uma das partes mais importantes deste trabalho é encontrar o modelo ARIMA ou SARIMA que melhor se ajuste a cada uma das séries temporais. Conceitos sobre séries temporais e a teoria por trás do processo de escolha do melhor modelo foi descrito em detalhes na Seção 2.2.

O R possui uma poderosa função chamada *auto.arima* [31], que retorna o melhor modelo ARIMA / SARIMA para séries temporais univariadas, utilizando como critério de seleção o menor valor AIC, AICc ou BIC. A função realiza uma pesquisa sobre o melhor modelo possível dentro das restrições de ordem fornecidas, utilizando o algoritmo *Stepwise* [52]. Combinando outros critérios, a utilização dessa função reduz significativamente o tempo computacional de escolha do modelo [53], principalmente para séries com muitas amostras ou para modelos sazonais complexos. Essa característica é essencial para este trabalho, já que o sistema precisa fazer ajustes e previsões em tempo real de várias séries de forma automatizada, e fornecer resultados em tempo hábil.

Assim, neste módulo do SDA a escolha do melhor modelo $SARIMA(p,d,q)(P,D,Q)_s$ é feita automaticamente através da função *auto.arima*. Por padrão, o número de diferenciações dado pelo termo d é calculado aplicando-se o Teste KPSS, enquanto que o cálculo do número de diferenciações sazonais D é feito a partir do Teste OCSB. A estimação dos termos autorregressivos p e P e dos termos de médias móveis q e Q utiliza por padrão

a função de máxima verossimilhança. Mais detalhes sobre o funcionamento da função *auto.arima* pode ser visto na sua documentação oficial [31].

As previsões em si são feitas utilizando-se a função *forecast* [31], uma função genérica do R que realiza predições em séries temporais usando os valores passados da série para prever valores futuros.

A Figura 3.6 mostra um gráfico de exemplo de ajuste de uma série temporal com o modelo SARIMA selecionado no R, juntamente com a previsão.

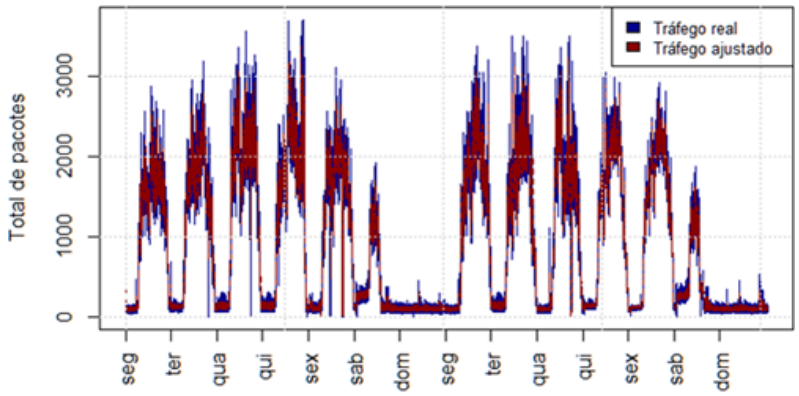


Figura 3.6: Série temporal UDP, ajustada com o modelo $SARIMA(4,1,5)(0,0,1)_{10}$

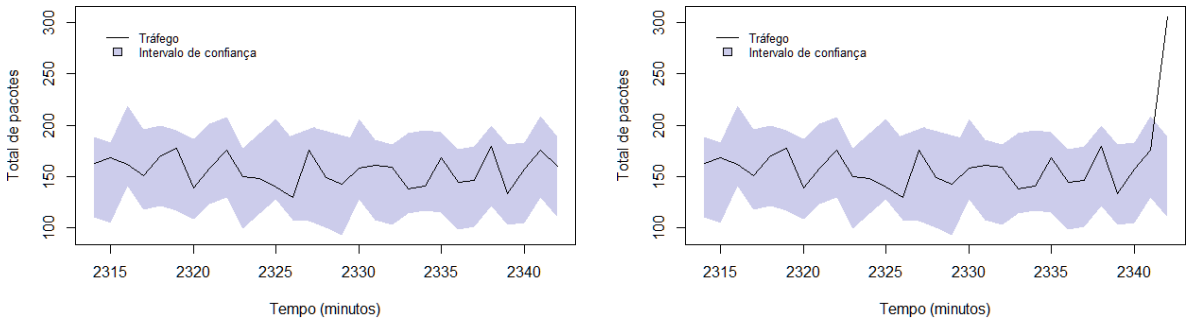
Identificação de Ataques

Após construir as séries temporais e encontrar o melhor modelo, é preciso caracterizar o comportamento do tráfego referente a cada série. Para tanto, o objetivo é classificar o valor das séries em um dado instante de tempo como normal ou anômalo. No caso deste estudo, a classificação de um valor como anômalo depende diretamente da definição do que seria o padrão de normalidade para um tráfego de pacotes. Por esse motivo é feita a modelagem com séries temporais. A definição de normalidade é feita através da análise do comportamento da série ao longo do tempo, considerando padrões como sazonalidade e tendência. Através da previsão do valor de uma série no próximo instante, baseado nos valores de períodos passados, é possível estimar qual seria o tráfego normal esperado para este instante de tempo. Um tráfego anormalmente intenso, que esteja acima do limite superior estabelecido para previsão nesse instante em particular, pode indicar uma anomalia decorrente de um ataque.

Neste trabalho, o limite superior considerado foi baseado no intervalo de confiança de 95% fornecido pala função *forecast* do R ao realizar previsões. Esse intervalo de

confiança foi escolhido por ter ser aplicado em alguns trabalhos relacionados utilizados como referência: [10, 23, 46, 54].

Um exemplo de detecção de anomalia na série temporal da métrica $ICMP$ é mostrado na Figura 3.7.



(a) Série temporal com valores medidos dentro do intervalo de confiança

(b) Ocorrência de anomalia no último valor medido da série temporal, que está acima do limite superior

Figura 3.7: Exemplo de detecção de anomalia em uma série temporal

Conforme foi explicado na Seção 3.3.2, cada um dos ataques de negação de serviço descritos na Seção 2.1.5 pode ser identificado a partir da detecção de uma anomalia em uma ou mais séries temporais correspondentes. Dessa forma, os ataques abordados neste trabalho são associados à ocorrência de anomalias da seguinte forma:

- *Fraggle Attack*: espera-se identificar esse tipo de ataque através da detecção de anomalias nas séries equivalentes às métricas UDP e $ICMP$. Nesse tipo de ataque, o atacante inunda um servidor com vários pacotes UDP, enquanto este tenta responder a cada requisição com pacotes ICMP para destinos inalcançáveis, gerando um tráfego anormalmente intenso desses dois tipos de pacote.

- *Smurf Attack*: espera-se identificar esse tipo de ataque através da detecção de anomalia na série equivalente à métrica $ICMP$. Ao inundar a vítima com vários pacotes ICMP, o atacante gera um tráfego muito alto desse tipo de pacote.

- *DNS Amplification*: espera-se identificar esse tipo de ataque através da detecção de anomalia na série equivalente às métricas UDP, $ICMP$ e DNS. As inúmeras solicitações DNS enviadas pelo atacante utilizam o UDP como protocolo da camada de transporte. A vítima, por sua vez, tenta responder a cada solicitação com pacotes ICMP. Por esse motivo, um ataque *DNS Amplification* causa um tráfego muito intenso de pacotes DNS, UDP e ICMP.

- *HTTP Floods*: espera-se identificar esse tipo de ataque através da detecção de anomalias nas séries equivalentes às métricas $HTTP$, TCP_ACK, TCP_SYN e TCP_SYN_ACK. Quando um atacante envia várias requisições HTTP para um servidor alvo, o protocolo utilizado na camada de transporte é o TCP. Como é necessário estabelecer uma conexão antes do envio de cada requisição, os dois *hosts* trocam pacotes TCP referentes ao *three-way handshake*. Portanto, durante esse ataque ocorrerá um tráfego intenso de pacotes HTTP e também de pacotes TCP com a $flag\ SYN$, com a $flag\ ACK$ e com as $flags\ SYN$ e ACK ativadas.

- *TCP SYN Flood*: espera-se identificar esse tipo de ataque através da detecção de anomalias nas séries equivalentes às métricas TCP_SYN, TCP_SYN_ACK e TCP_RST_ACK. Quando o atacante inunda a vítima com um grande número de pacotes TCP SYN, esta tenta continuar o estabelecimento das conexões enviando vários pacotes TCP com as $flags\ SYN$ e ACK. Como as tentativas de conexão não são bem sucedidas, a vítima as rejeita enviando muitos pacotes TCP com as $flags\ RST$ e ACK ativadas. Por esse motivo, um ataque *TCP SYN Flood* gera um grande tráfego desses três tipos de pacote.

- *TCP Reset Attack*: espera-se identificar esse tipo de ataque através da detecção de anomalia na série equivalente à métrica TCP_RST, uma vez que o atacante envia um número muito grande de pacotes TCP que possuem somente a $flag\ RST$ ativa para realizar esse ataque, e não recebe nenhum pacote como resposta do *host* alvo.

- *TCP PSH+ACK Flood*: espera-se identificar esse tipo de ataque através da detecção de anomalia na série equivalente às métricas TCP_RST e TCP_PSH_ACK. Isso se deve ao fato de o atacante gerar um tráfego anormalmente intenso de pacotes TCP falsos com as $flags\ PSH$ e ACK para realizar esse ataque, que são respondidos pela vítima com vários pacotes TCP RST, na tentativa de reiniciar as comunicações inválidas.

- *TCP FIN+ACK Flood*: espera-se identificar esse tipo de ataque através da detecção de anomalia na série equivalente à métrica TCP_FIN_ACK. Nesse ataque, o atacante causa um tráfego muito alto de pacotes TCP com as $flags\ FIN$ e ACK, que não são respondidos pela vítima.

- *Spoofed Session Attack*: espera-se identificar esse tipo de ataque através da detecção de anomalia na série equivalente às métricas TCP_ACK e TCP_RST, pois ao realizar esse ataque, o atacante gera um grande tráfego de pacotes TCP ACK,

que são respondidos pelo alvo com inúmeros pacotes TCP *RST* para recusar esses pacotes inválidos.

A Tabela 3.1 resume a relação de cada ataque de negação de serviço com cada métrica principal, com as associações explicadas anteriormente.

	UDP	*ICMP*	*DNS*	*HTTP*	*TCP_ACK*	*TCP_SYN*	*TCP_SYN_ACK*	*TCP_RST*	*TCP_RST_ACK*	*TCP_PSH_ACK*	*TCP_FIN_ACK*
Fraggle Attack	*	*									
Smurf Attack		*									
DNS Amplification	*	*	*								
HTTP Floods				*	*	*	*				
TCP SYN Flood						*	*		*		
TCP Reset Attack								*			
TCP PSH+ACK Flood								*		*	
TCP FIN+ACK Flood											*
Spoofed Session Attack					*			*			

Tabela 3.1: Associação das métricas principais com os ataques de negação de serviço

De forma resumida, a ideia geral de detecção de um ataque é descrita a seguir:

1. Encontrar o modelo ARIMA / SARIMA que melhor se ajuste a cada série temporal.
2. Utilizar o modelo encontrado para efetuar a previsão do valor da série no instante de tempo t, com base nos valores até o instante de tempo $t-1$.
3. Definir um limite superior baseado no valor previsto para o instante de tempo t.
4. Analisar o valor real observado no instante de tempo t. Se esse valor se encontrar acima do limite superior esperado, significa que pode estar ocorrendo uma anomalia no tráfego devido a um ataque.
5. Para cada tipo de ataque, se ocorreram anomalias nas séries referentes às métricas relacionadas, enviar um alerta por e-mail para o gerente de rede, informando o momento de detecção e qual o ataque que pode estar ocorrendo. Enviar também um aviso por e-mail em caso de ocorrência de anomalias sem identificação de um ataque específico.

O pseudo-código do Algoritmo 1 ilustra a ideia de detecção de anomalias e identificação de ataques, referentes ao terceiro módulo do SDA. Para esse algoritmo, são considerados como dados de entrada:

- os valores obtidos para cada métrica no módulo de seleção de dados;
- a granularidade de 1 minuto;
- as restrições de ordem para seleção dos modelos de previsão;
- o intervalo de confiança de 95%;
- uma matriz que relaciona cada tipo de ataque de negação de serviço com cada uma das métricas principais.

Algoritmo 1 Detecção de anomalias e identificação de ataques no SDA

```
1:  para cada métrica faça
2:     construirSérieTemporal()
3:     selecionarMelhorModelo()
4:     até instante t-1
5:        ajustarSerieAoModelo()
6:     fim até
7:     fazerPrevisao()
8:     definirLimiteSuperior()
9:     se valor_real > limite_superior então
10:       anomalia ← TRUE
11:       emitirAlerta()
12:    senão
13:       anomalia ← FALSE
14:    fim se
15: fim para
16: para cada tipo de ataque faça
17:    se métricas correspondentes apresentaram anomalia então
18:       emitirAlerta()
19:    fim se
20: fim para
```

Capítulo 4

Resultados e Discussões

Este capítulo apresenta análises e discussões sobre os resultados obtidos com a aplicação prática da metodologia proposta neste trabalho na rede do IFRJ - Campus Volta Redonda, que se divide basicamente em três etapas. A primeira etapa corresponde a uma análise dos dados capturados e das métricas principais propostas. Na segunda etapa foram avaliados os modelos de previsão selecionados com a função *auto.arima* para as séries temporais. Na terceira etapa foram feitas simulações de ataques de negação de serviço, a fim de validar a detecção desses ataques pelo SDA. Neste capítulo também são apresentadas discussões acerca das principais contribuições do trabalho para o cenário real de estudo.

4.1 Análise das Métricas

4.1.1 Dados Capturados

Para a verificação das métricas propostas neste trabalho, foram analisados os dados referentes ao tráfego capturado entre os dias 07/05/2018 e 20/05/2018, totalizando um período de duas semanas. Como pré-condições de monitoramento do ambiente de estudo, é importante salientar que:

- Esse período foi escolhido por apresentar um comportamento típico e normal da utilização dos serviços de rede e Internet no campus em dias letivos. Nessas duas semanas não houve feriados ou recessos escolares e não ocorreram quedas de conexão nem falhas nos principais servidores e equipamentos de rede;

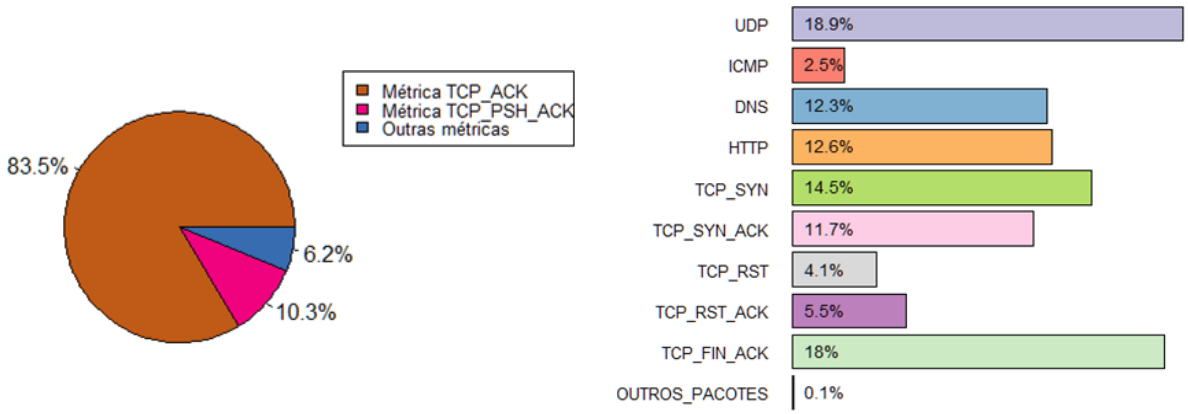
- A instituição segue rígidos parâmetros de segurança em redes, através da utilização de anti-vírus, *firewall* e outros ferramentas de proteção, com o objetivo de garantir a segurança e a integridade dos dados trafegados e dos recursos computacionais utilizados pelos usuários.

Além disso, o SDA não detectou nenhuma anomalia no tráfego nem nenhum ataque durante essas duas semanas. Dessa forma, pode-se considerar que o tráfego observado nesse período está isento de atividades anormais e maliciosas, sendo possível caracterizar e analisar um tráfego comum, estabelecendo-se a definição de um comportamento padrão e normal do uso da rede, considerando os períodos com maior atividade por parte dos usuários. Nos meses anteriores, foram feitos testes iniciais também com outros períodos de captura, mas os resultados obtidos não foram satisfatórios, pois a metodologia ainda estava em fase de teste, no que diz respeito à definição das métricas e como e quais ataques seriam estudados. Além disso, ocorreram falhas no processo de captura e quedas de conexão com alguns importantes serviços de rede durante essa fase de teste.

Conforme proposto na Seção 3.3, foram definidas onze métricas referentes ao número total de um tipo específico de pacote transmitido pela rede, e mais duas métricas que correspondem ao total de outros pacotes e ao total de todos os pacotes, respectivamente. Como a captura dos dados foi feita a cada 1 minuto no período de duas semanas, foi obtido um total de 20160 amostras para cada série temporal. Durante esse período de duas semanas considerado para a análise, houve pequenas falhas no processo de seleção de dados. Para cada série temporal, ocorreram em média 8 medições duplicadas e 3 ausentes. Essas falhas não influenciaram no desempenho e funcionamento do método, mas cabe em um trabalho futuro o estudo de uma melhoria na etapa de captura de dados. Uma possível causa dessas eventuais medições incorretas pode ser atribuída a falhas de *hardware*. Para contornar esse problema, pretende-se substituir futuramente o servidor SDA atual por um computador mais robusto. Outra possível melhoria é a utilização de outros computadores conectados a outras interfaces espelhadas do roteador central, de forma a realizar a captura do tráfego de maneira simultânea e redundante, a fim de evitar problemas relacionados à medições duplicadas ou ausentes.

4.1.2 Comparações entre as Métricas

Para comparar as métricas entre si, foi feita uma análise do número total de cada tipo de pacote que trafegou na rede durante o período, além do total de outros pacotes. Os resultados são apresentados na Figura 4.1.



(a) TCP_ACK, TCP_PSH_ACK e outras métricas

(b) Outras métricas

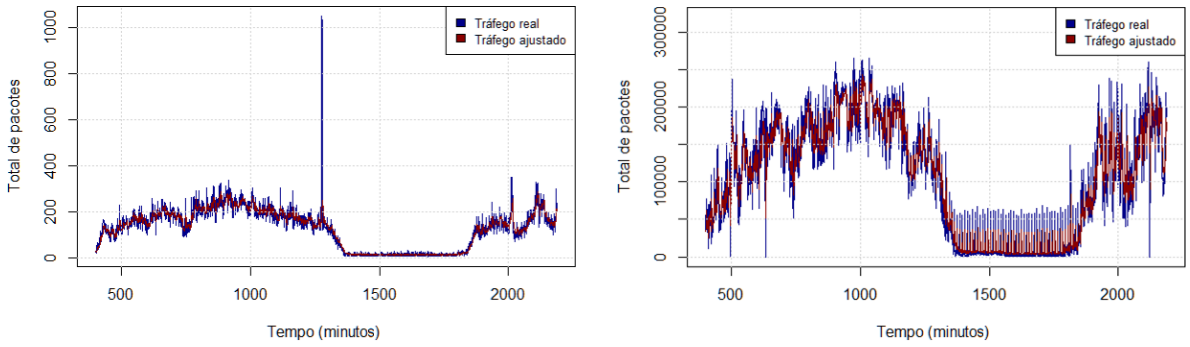
Figura 4.1: Pacotes trafegados referentes a cada métrica

De acordo com o primeiro gráfico na Figura 4.1(a), foi observada uma grande discrepância entre o número de pacotes da métrica TCP_ACK em relação às outras métricas, e também um número relativamente grande de pacotes da métrica TCP_PSH_ACK. Esse resultado já era esperado, uma vez que a maioria das aplicações da Internet utiliza o TCP como protocolo na camada de transporte, e pacotes TCP que possuem somente a *flag* ACK ativa são muito mais comuns que outros pacotes TCP durante as comunicações entre *hosts*.

A métrica $OUTROS\_PACOTES$ corresponde aos muitos outros protocolos utilizados nas cinco camadas do modelo TCP/IP, cujos pacotes também foram capturados, porém não foram abordados neste trabalho. Apesar de essa métrica ter apresentado um número muito pouco significativo de pacotes, ela é importante para a detecção de outras possíveis anomalias que não estejam associadas a nenhum dos ataques de negação de serviço considerados.

A análise do segundo gráfico na Figura 4.1(b) ressalta a importância da utilização das métricas principais propostas para análise do tráfego da rede. Se fosse analisada somente a série temporal referente à métrica $TOTAL\_PACOTES$, por exemplo, os ataques gerariam anomalias muito mais diluídas, dificultando a detecção. Além disso, não seria possível diferenciar os tipos de ataque somente com essa análise. Por esse motivo, as métricas que apresentam um menor número de pacotes trafegados são muito importantes para a detecção de ataques específicos. Esse fato pode ser confirmado pela simples observação dos gráficos da Figura 4.2.

O primeiro gráfico na Figura 4.2(a) mostra a série temporal referente à métrica



(a) Métrica $ICMP$ (b) Métrica $TOTAL\_PACOTES$

Figura 4.2: Séries temporais com ocorrência de anomalia

$ICMP$, plotada juntamente com os ajustes realizados com o SDA. Essa métrica corresponde a somente 0,15% do total de pacotes trafegados (2,5% dos 6,2% referentes às métricas diferentes de TCP_ACK e TCP_PSH_ACK). Próximo ao instante de tempo 1250, é visualmente perceptível que ocorreu uma anomalia devido a um tráfego intenso de pacotes ICMP, que foi induzida através de um teste. O segundo gráfico, na Figura 4.2(b), corresponde à série temporal da métrica $TOTAL\_PACOTES$ e o seu respectivo ajuste. Nesse gráfico a mesma anomalia não causa uma mudança significativa no comportamento da série, uma vez que acréscimo de aproximadamente 1.000 pacotes ICMP não influencia no tráfego total, que no mesmo instante é da ordem de 200.000 pacotes. Os ajustes e previsões feitos para cada métrica são avaliados em detalhes na Seção 4.2, e na Seção 4.3 é descrito o procedimento de indução de anomalias.

4.2 Avaliação dos Modelos

4.2.1 Seleção Computacional

Conforme abordado na Seção 3.3.3, neste trabalho a seleção dos modelos de previsão é feita computacionalmente utilizando a função *auto.arima* do R, que escolhe o menor valor AIC, AICc ou BIC como critério de seleção do melhor modelo. Os modelos selecionados pelo SDA para cada série temporal são mostrados na Tabela 4.1, juntamente com os respectivos valores AIC, AICc e BIC obtidos. Foram calculados também o MAPE do ajuste, correspondente ao ajuste de toda a série, e o MAPE da previsão, referente somente ao valor previsto para o último instante de tempo em relação ao último valor real. Para incluir modelos SARIMA na seleção de modelos, o termo s (período de sazonalidade) foi definido como 10. Esse valor foi obtido através do número de defasagens (*lags*) das

funções de autocorrelação e autocorrelação parcial das séries, conforme exemplificado nas Figuras 4.5, 4.6 e 4.7.

Série temporal / métrica	Modelo selecionado	AIC	AICc	BIC	MAPE do ajuste (%)	MAPE da previsão (%)	Tempo (s)
UDP	$SARIMA(4,1,5)(0,0,1)_{10}$	269980,9	269980,9	270067,9	28,73	28,57	0,150000
ICMP	$ARIMA(1,1,2)$	170344,1	170344,1	170375,8	19,49	19,40	0,009801
DNS	$SARIMA(3,1,4)(0,0,2)_{10}$	255962,0	255962,1	256041,2	30,15	12,79	0,050007
HTTP	$SARIMA(2,1,3)(0,0,2)_{10}$	340959,8	340959,8	341023,1	33,92	24,73	0,109901
TCP_ACK	$SARIMA(3,1,3)(2,0,0)_{10}$	454022,0	454022,0	454093,2	27,02	23,25	0,039991
TCP_SYN	$SARIMA(4,1,3)(0,0,2)_{10}$	300250,8	300250,8	300329,9	24,57	10,97	0,060023
TCP_SYN_ACK	$SARIMA(1,1,2)(0,0,1)_{10}$	272381,6	272381,6	272421,1	23,85	16,25	0,009981
TCP_RST	$SARIMA(2,1,2)(0,0,2)_{10}$	238634,1	238634,1	238689,5	27,50	21,62	0,009801
TCP_RST_ACK	$SARIMA(2,1,4)(0,0,2)_{10}$	282137,4	282137,5	282208,6	32,68	20,65	0,019996
TCP_PSH_ACK	$ARIMA(4,1,3)$	358953,9	358953,9	359017,2	17,74	12,28	0,019996
TCP_FIN_ACK	$ARIMA(2,1,2)$	280847,9	280847,9	280887,5	19,01	21,98	0,039991
OUTROS_PACOTES	$SARIMA(0,1,1)(1,0,2)_{10}$	124106,9	124106,9	124146,4	30,00	24,50	0,001890
TOTAL_PACOTES	$SARIMA(5,1,3)(2,0,0)_{10}$	454986,0	454986,0	455073,0	26,46	24,63	0,100790
						Tempo total (s)	0,622168

Tabela 4.1: Modelos selecionados para cada série temporal com a função *auto.arima*, referente ao período de 07/05/2018 às 00:00 até 20/05/2018 às 23:59

Os valores do MAPE para todas as séries temporais, tanto do ajuste quanto da previsão, estão abaixo de 35%, que é um valor aceitável para previsões neste trabalho, segundo comparação feita com outros trabalhos com abordagem similar, a saber: [6, 43, 56]. Quanto mais baixo o MAPE, mais precisa é uma previsão. Segundo a autora de [56], pode-se interpretar o MAPE julgando a previsão da seguinte maneira: com menos de 10% a previsão é altamente precisa; de 11% a 20% é uma boa previsão; de 20% a 50% é uma previsão razoável; e acima 50% a previsão é imprecisa. Para o objetivo do presente trabalho, segundo os valores de MAPE, os modelos selecionados são suficientemente precisos para comparar previsões com o tráfego real, uma vez que os ataques estudados geram tráfegos notadamente intensos para as métricas consideradas, segundo os testes realizados. Os resultados das simulações de ataques são detalhados na Seção 4.3.

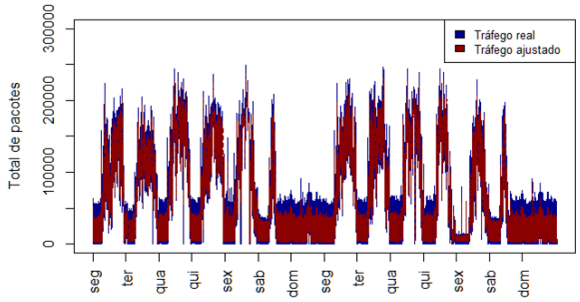
Considerando que a identificação de ataques deve ser feita *online*, dentro do intervalo de 1 minuto no qual os dados são capturados, a eficiência de cada execução de uma instância do SDA é um fator crítico para o desempenho do método. A análise dessa eficiência pode ser feita experimentalmente, testando o programa na prática e medindo o tempo de execução computacional. Ao medir esse tempo para cada módulo do sistema,

percebeu-se que os módulos de captura de tráfego e seleção de dados possuem tempos de execução baixíssimos, sendo irrelevantes para o tempo total. Por outro lado, o módulo de detecção de anomalias possui um tempo de execução mais significante no trecho referente à seleção automática e ajuste de modelos. Assim sendo, o tempo computacional mostrado na Tabela 4.1 se refere especificamente ao tempo de execução da função *auto.arima*. Os tempos em segundos apresentados na tabela correspondem a medidas do tempo de CPU feitas no próprio módulo do sistema, com a função *proc.time* do R [31], sendo que os valores obtidos equivalem a uma média dos resultados de 10 testes de execução no servidor SDA. O tempo de CPU, ou tempo de processamento, se refere à quantidade de tempo em que a CPU foi usada para processar as instruções do trecho do programa em questão, considerando apenas o tempo em que o processo ficou de fato executando no processador.

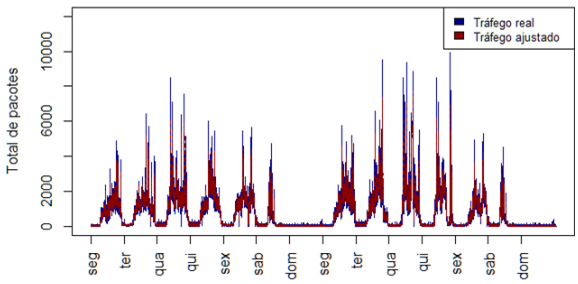
Outra forma de avaliar a eficiência do SDA é verificando a complexidade do algoritmo que implementa os ajustes e previsões nos modelos paramétricos ARIMA / SARIMA. Segundo os autores do trabalho [57], a complexidade computacional para esses modelos é $O(n^2)$. Existem outros modelos diferentes encontrados na literatura aplicados para previsão de tráfego em redes, que utilizam técnicas diferentes de análise de séries temporais. Alguns desses modelos possuem menor complexidade, como por exemplo, os baseados em redes neurais artificiais. Modelos de séries temporais baseados em ARIMA são mais precisos, apesar de complexos computacionalmente, apresentando excelentes resultados para previsão de tráfego em redes, tanto para tráfego de curta dependência, quanto de longa dependência [6, 7, 57].

Com os testes experimentais de medição do tempo de execução do SDA, pode-se concluir que a complexidade dos modelos não é um fator tão relevante para o desempenho do método proposto neste trabalho, uma vez que o tempo total de execução do módulo de detecção de anomalias, apesar de significante, não é suficiente para influenciar negativamente no desempenho do método, e consequentemente, no correto funcionamento do SDA. De acordo com os resultados apresentados na Tabela 4.1, O tempo computacional total necessário para seleção e ajuste dos modelos está na faixa de 0,62 segundos, que é um valor consideravelmente baixo, uma vez que todo o processo de execução do sistema deve se dar em até 1 minuto.

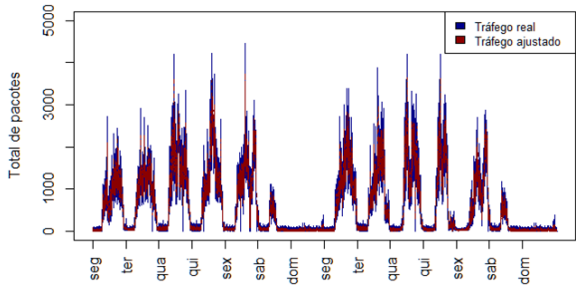
As Figuras 4.3, e 4.4 contêm os gráficos dos ajustes obtidos para todas as séries temporais utilizando os respectivos modelos de previsão selecionados computacionalmente, no período de duas semanas considerado.



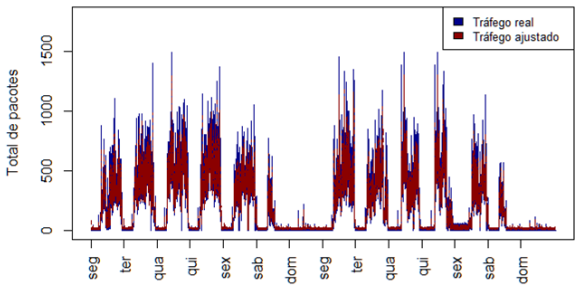
(a) TCP_ACK: SARIMA(3, 1, 3)(2, 0, 0)$_{10}$



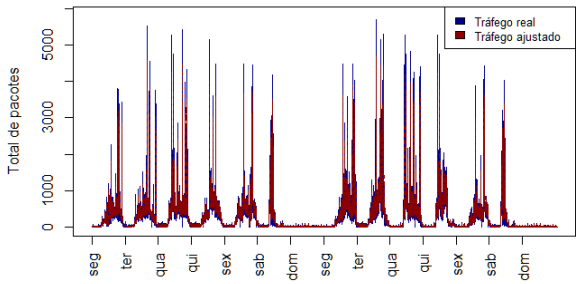
(b) TCP_SYN: SARIMA(4, 1, 3)(0, 0, 2)$_{10}$



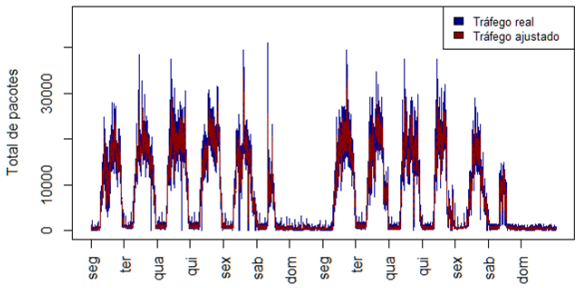
(c) TCP_SYN_ACK: SARIMA(1, 1, 2)(0, 0, 1)$_{10}$



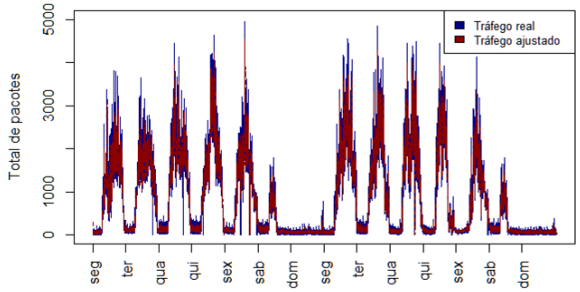
(d) TCP_RST: SARIMA(2, 1, 2)(0, 0, 2)$_{10}$



(e) TCP_RST_ACK: SARIMA(2, 1, 4)(0, 0, 2)$_{10}$

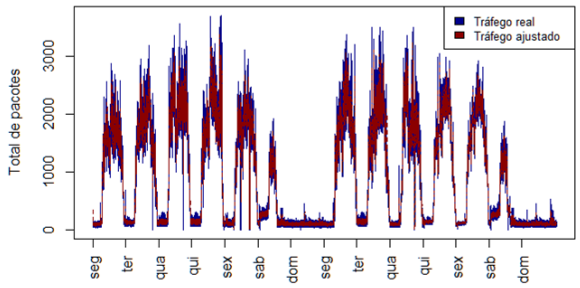


(f) TCP_PSH_ACK: ARIMA(4, 1, 3)

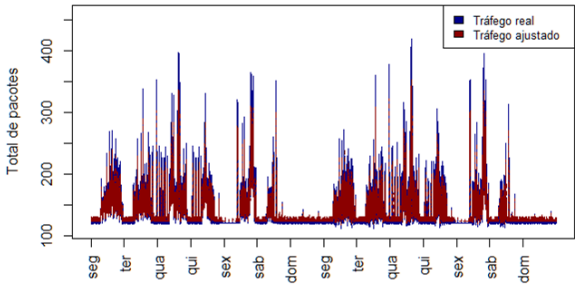


(g) TCP_FIN_ACK: ARIMA(2, 1, 2)

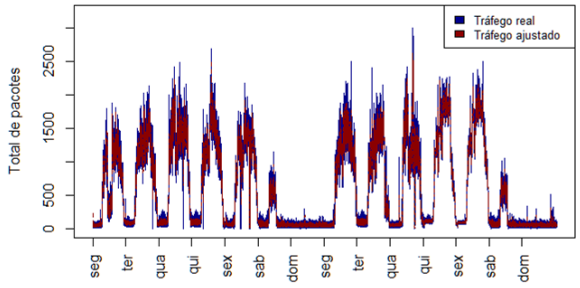
Figura 4.3: Séries temporais das métricas referentes ao protocolo TCP, com os respectivos ajustes obtidos com os modelos selecionados computacionalmente



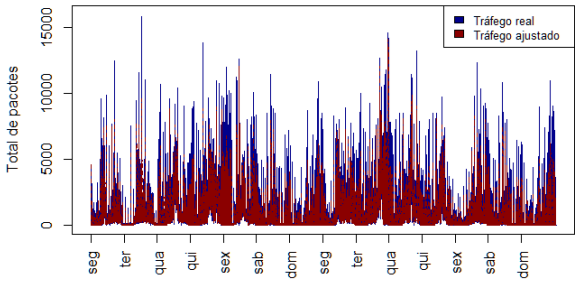
(a) *UDP*: SARIMA(4, 1, 5)(0, 0, 1)$_{10}$



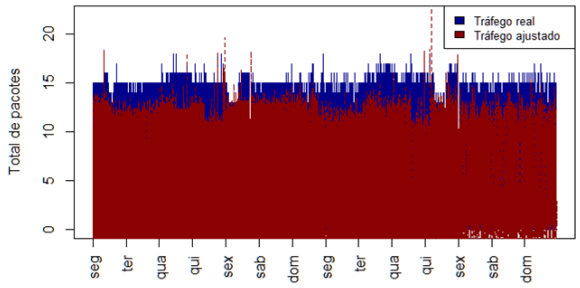
(b) *ICMP*: ARIMA(1, 1, 2)



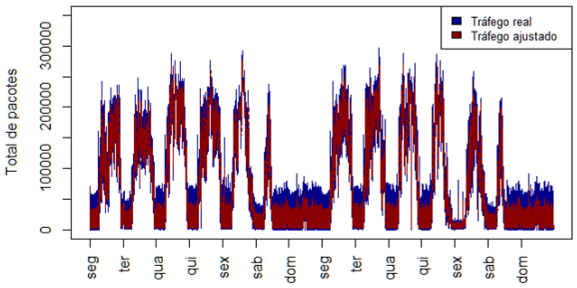
(c) *DNS*: SARIMA(3, 1, 4)(0, 0, 2)$_{10}$



(d) *HTTP*: SARIMA(2, 1, 3)(0, 0, 2)$_{10}$



(e) *OUTROS_PACOTES*: SARIMA(0, 1, 1)(1, 0, 2)$_{10}$

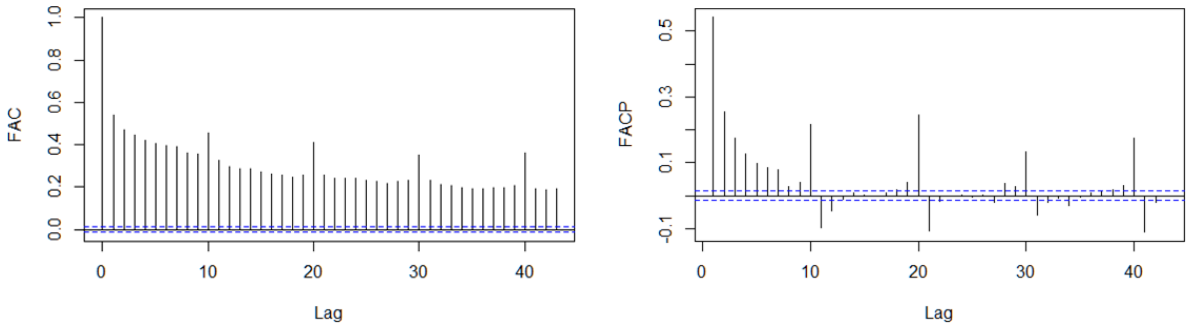


(f) *TOTAL_PACOTES*: SARIMA(5, 1, 3)(2, 0, 0)$_{10}$

Figura 4.4: Séries temporais *UDP*, *ICMP*, *DNS*, *HTTP*, *OUTROS_PACOTES* e *TOTAL_PACOTES*, com os respectivos ajustes obtidos com os modelos selecionados computacionalmente

4.2.2 Seleção Manual

Como forma de validar a seleção de modelos feita pelo SDA com a função *auto.arima*, as séries temporais foram analisadas e a identificação dos modelos foi feita de forma manual, baseando a estimação dos termos dos modelos na análise gráfica das funções de autocorrelação e autocorrelação parcial, conforme explicado na Seção 2.2.3. Na Figura 4.5 são mostradas a função de autocorrelação (FAC) e a função de autocorrelação parcial (FACP) para a série temporal referente à métrica $HTTP$.



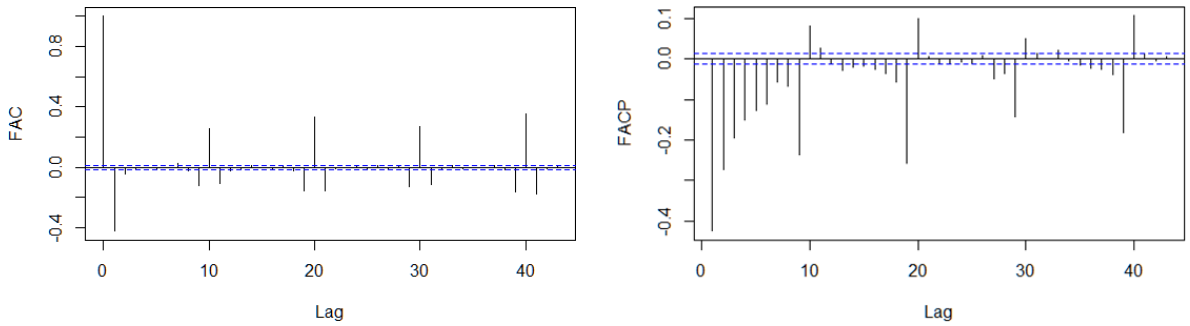
(a) Função de autocorrelação

(b) Função de autocorrelação parcial

Figura 4.5: Funções de autocorrelação e autocorrelação parcial da série temporal da métrica $HTTP$

A Figura 4.5(a) apresenta o gráfico da função de autocorrelação da série temporal. Através desse gráfico, já é possível perceber que a série não é estacionária, dado o decaimento lento da FAC. Esse comportamento é um indicativo de que a série apresenta tendência. Através do gráfico da FAC, é possível concluir também que a série apresenta sazonalidade, devido à presença de picos sazonais a cada 10 defasagens.

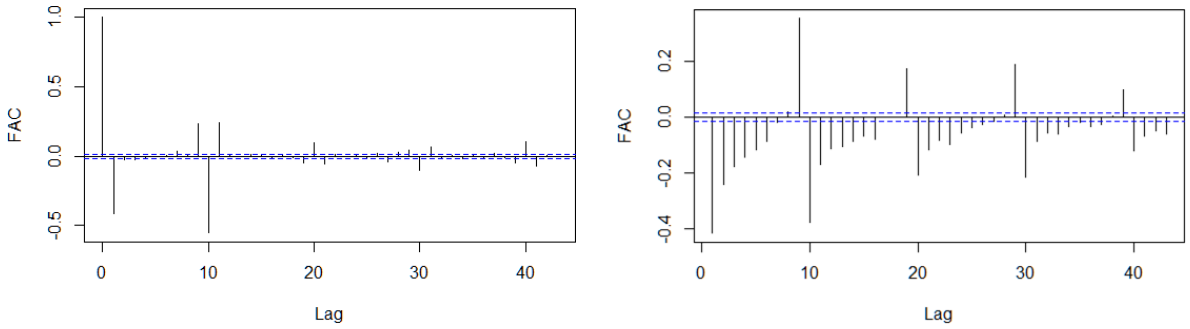
Para eliminar a tendência, é feita uma diferenciação na série, obtendo-se assim o termo $d = 1$. Os gráficos das funções de autocorrelação e autocorrelação parcial da série diferenciada são mostrados na Figura 4.6.



(a) Função de autocorrelação (b) Função de autocorrelação parcial

Figura 4.6: Funções de autocorrelação e autocorrelação parcial da série temporal da métrica $HTTP$ após uma diferenciação

De acordo com a Figura 4.6(a), é possível observar a presença de picos no gráfico da FAC, nos *lags* sazonais $h = 10, h = 20, h = 30, h = 40$ e assim sucessivamente, caracterizando um comportamento de sazonalidade. Sugere-se então que seja feita uma diferenciação sazonal com período de sazonalidade $s = 10$, obtendo-se $D = 1$. Os gráficos referentes a essa segunda diferenciação são mostrados na Figura 4.7.



(a) Função de autocorrelação (b) Função de autocorrelação parcial

Figura 4.7: Funções de autocorrelação e autocorrelação parcial da série temporal da métrica $HTTP$ após uma diferenciação e uma diferenciação sazonal

O gráfico da FAC da Figura 4.7(a) não apresenta decaimento lento em nenhum *lag*, portanto não é mais necessária nenhuma diferenciação para tornar a série estacionária. Como $s = 10$, observam-se os nove primeiros *lags* da FAC e da FACP para obter os termos não sazonais q e p, respectivamente. Os termos sazonais Q e P, por sua vez, são obtidos através da observação de picos nos *lags* sazonais (a partir de $h = 10$) da FAC e da FACP.

É possível notar a ocorrência de três picos não sazonais no gráfico da FAC, em $h = 0$, $h = 1$ e $h = 9$. Assim sendo, o termo não sazonal de médias moveis é $q = 3$. Na parte

sazonal, existe apenas um pico significativo, em $h = 10$, o que sugere o termo sazonal de médias móveis como $Q = 1$.

No gráfico da FACP, presente na Figura 4.7(b), ocorrem alguns picos não sazonais. Ao considerar apenas os picos mais significativos ($h = 1$ e $h = 9$), obtém-se o termo autorregressivo não sazonal $p = 2$. A FACP apresenta também picos nos quatro primeiros *lags* sazonais, com comportamento decrescente. Pode considerar então o termo autorregressivo sazonal como $P = 4$.

De acordo com essa análise, o modelo obtido é o $SARIMA(2,1,3)(4,1,1)_{10}$. Esse modelo é diferente daquele obtido para a série temporal da métrica $HTTP$ com a função *auto.arima*, de acordo com a Tabela 4.1, que é o $SARIMA(2,1,3)(0,0,2)_{10}$. Os termos da parte não sazonal (p, d e q) são os mesmos, mas a parte sazonal apresenta um menor número de parâmetros, e, além disso, não é feita nenhuma diferenciação sazonal.

Calculando os valores dos critérios de seleção para o modelo obtido com a análise gráfica da FAC e da FACP, obtém-se AIC = AICc = 342586,8 e BIC = 342650,1. Esses valores são um pouco maiores do que aqueles obtidos com o modelo selecionado pela função *auto.arima*: AIC = AICc = 340959,8 e BIC = 341023,1.

Foram calculados também o MAPE do ajuste e o MAPE referente à previsão, obtendo-se, respectivamente, 54,95% e 54,81%. Esses valores, além de serem maiores do que o aceitável para uma boa previsão, são superiores aos valores de 31,92% e 24,73% calculados com o modelo selecionado pela função *auto.arima*.

Portanto, a escolha mais parcimoniosa para a série temporal da métrica $HTTP$ é o modelo $SARIMA(2,1,3)(0,0,2)_{10}$. Assim como para a métrica $HTTP$, foi feita uma análise gráfica das funções de autocorrelação e autocorrelação parcial também para as séries temporais das outras métricas, a fim de selecionar os modelos manualmente. A Tabela 4.2 apresenta os resultados referentes a essa análise.

Comparando os resultados das Tabelas 4.1 e 4.2, é possível observar que quase todos os modelos selecionados computacionalmente com a função *auto.arima* apresentam valores um pouco menores tanto para o AIC, quanto para o AICc e para o BIC, em relação aos resultados obtidos manualmente com a análise gráfica da FAC e da FACP. Os valores do MAPE obtidos com o *auto.arima* também são melhores no geral, sendo que quatro métricas apresentaram MAPE acima de 50% com a seleção manual de modelos. Essa análise comparativa é suficiente para considerar a escolha dos modelos selecionados computacionalmente como a mais adequada para este trabalho. Mesmo para as métricas

Série temporal / métrica	Modelo selecionado	AIC	AICc	BIC	MAPE do ajuste (%)	MAPE da previsão (%)
UDP	$ARIMA(4,1,2)$	270054,6	270054,6	270110,0	32,23	16,90
$ICMP$	$ARIMA(3,1,2)$	170311,3	170311,3	170358,8	16,94	21,18
DNS	$SARIMA(4,1,2)(0,0,1)_{10}$	255726,9	255726,9	255806,0	49,02	13,14
$HTTP$	$SARIMA(2,1,3)(4,1,1)_{10}$	342586,8	342586,8	342650,1	54,95	54,81
TCP_ACK	$SARIMA(3,1,3)(4,1,1)_{10}$	461223,7	461223,7	461294,9	48,70	17,54
TCP_SYN	$SARIMA(5,1,3)(1,0,0)_{10}$	300367,5	300367,5	300430,8	54,17	15,29
TCP_SYN_ACK	$ARIMA(4,1,2)$	272411,9	272411,9	272443,6	54,98	14,47
TCP_RST	$ARIMA(3,1,2)$	238667,4	238667,4	238714,9	47,31	52,28
TCP_RST_ACK	$SARIMA(3,1,1)(1,0,1)_{10}$	282139,8	282139,8	282203,1	49,79	15,07
TCP_PSH_ACK	$ARIMA(4,1,2)$	358965,4	358965,5	359020,8	32,84	10,70
TCP_FIN_ACK	$ARIMA(3,1,2)$	280873,9	280873,9	280921,4	43,93	20,00
$OUTROS\_PACOTES$	$ARIMA(2,1,2)$	111671,8	111671,8	111711,4	30,50	10,50
$TOTAL\_PACOTES$	$SARIMA(3,1,1)(4,1,1)_{10}$	451617,0	451617,0	451696,1	32,45	14,24

Tabela 4.2: Modelos selecionados manualmente, segundo análise da FAC e da FACP, referente ao período de 07/05/2018 às 00:00 até 20/05/2018 às 23:59

que apresentaram valores um pouco menores para os critérios de seleção com os modelos selecionados manualmente, essa pequena diferença não justifica a não utilização da implementação computacional proposta com o SDA para todas as métricas.

4.2.3 Considerações Adicionais

Antes de se iniciar o processo de detecção de anomalias e identificação de ataques, é necessário o estabelecimento de um comportamento padrão para as séries temporais de cada métrica. Assim, inicialmente foi feita a captura do tráfego e a obtenção dos valores das métricas referentes a um período de duas semanas, para então, a seguir, iniciar-se a rotina de verificação correspondente ao terceiro módulo do SDA.

Com o objetivo de melhorar o desempenho do sistema e garantir a agilidade da detecção de anomalias, as previsões feitas a cada 1 minuto são baseadas nas respectivas séries temporais referentes sempre ao período de duas semanas anteriores, somente. Dessa forma, as séries apresentam comportamento dinâmico, alterando-se o início e o fim das amostras a cada ajuste e previsão, sempre com uma janela de duas semanas.

A Tabela 4.3 apresenta os resultados obtidos com a seleção de modelos com o função *auto.arima* para as séries temporais com amostragens iniciando-se em 09/05/2018 às 15:00 e terminando em 23/05/2018 às 14:59.

Série temporal / métrica	Modelo selecionado	AIC	AICc	BIC	MAPE do ajuste (%)	MAPE da previsão (%)	Tempo (s)
UDP	$SARIMA(1,1,3)(2,0,0)_{10}$	216945,6	216945,6	216999,5	30,43	25,56	0,010000
$ICMP$	$SARIMA(3,1,1)(2,0,1)_{10}$	138041,6	138041,6	138103,2	20,69	12,14	0,030011
DNS	$SARIMA(3,1,3)(2,0,1)_{10}$	206080,4	206080,4	206157,4	32,50	19,58	0,060001
$HTTP$	$SARIMA(2,1,3)(0,0,2)_{10}$	275721,9	275721,9	275783,6	29,24	15,44	0,050000
TCP_ACK	$SARIMA(4,1,3)(2,0,0)_{10}$	367619,9	367619,9	367696,9	22,79	26,85	0,020001
TCP_SYN	$SARIMA(5,1,5)(1,0,1)_{10}$	243059,3	243059,3	243159,4	23,56	9,71	0,049998
TCP_SYN_ACK	$SARIMA(4,1,4)(2,0,0)_{10}$	219402,2	219402,2	219487,0	25,10	14,03	0,210000
TCP_RST	$SARIMA(2,1,2)(1,0,0)_{10}$	191995,8	191995,8	192042,0	28,97	21,30	0,020000
TCP_RST_ACK	$SARIMA(3,1,3)(0,0,1)_{10}$	227563,6	227563,6	227625,2	31,71	22,61	0,029998
TCP_PSH_ACK	$SARIMA(2,1,2)(2,0,2)_{10}$	290238,1	290238,1	290315,1	19,27	11,30	0,049989
TCP_FIN_ACK	$ARIMA(2,1,4)$	280808,2	280808,2	280863,6	23,44	19,99	0,018990
$OUTROS\_PACOTES$	$ARIMA(2,1,2)$	111671,8	111671,8	111711,4	30,50	12,00	0,002000
$TOTAL\_PACOTES$	$SARIMA(5,1,3)(2,0,0)_{10}$	454986,0	454986,0	455073,0	26,46	18,04	0,099070
						Tempo total (s)	0,650058

Tabela 4.3: Modelos selecionados para cada série temporal com a função *auto.arima*, referente ao período de 09/05/2018 às 15:00 até 23/05/2018 às 14:59

Como já era de se esperar, os modelos selecionados nesse caso podem ser diferentes, dada a mudança de estrutura das séries. É preciso então implementar uma seleção dinâmica do melhor modelo, o que reforça a necessidade de uma aplicação computacional para essa finalidade. Os valores de MAPE calculados para os modelos da Tabela 4.3 também encontram-se todos abaixo de 35%, e o tempo computacional total não é suficiente para influenciar negativamente no desempenho do SDA.

4.3 Simulação de Ataques

4.3.1 Preparação do Ambiente de Testes

Durante a fase de teste da metodologia, observou-se que anomalias muito discrepantes prejudicavam a qualidade das previsões feitas com os modelos ARIMA / SARIMA para as séries temporais, e consequentemente, a capacidade de detecção de anomalias. Isso se deve ao fato de o modelo de previsão considerar os valores anômalos como parte do comportamento normal do tráfego. Por esse motivo, o sistema foi modificado para que os ajustes em cada série temporal sejam feitos sem considerar as anomalias detectadas. Assim sendo, sempre que o SDA detecta uma anomalia, ele não considera esse valor anômalo para fazer a previsão para o próximo instante. Em vez disso, ele utiliza a própria

previsão feita para o instante atual para compor a série temporal e prever o próximo valor.

Com o objetivo de validar a metodologia proposta no presente trabalho, foram feitos testes de indução de anomalias, através de simulações de diferentes ataques de negação de serviço na rede do IFRJ - Campus Volta Redonda. Os ataques foram realizados primeiramente de maneira individual, isto é, executando-se um ataque diferente por vez, utilizando uma máquina virtual Kali Linux [58]. O Kali Linux é uma distribuição Linux que contém várias ferramentas nativas voltadas para testes de invasão, ataques, e análise de vulnerabilidades, sendo amplamente utilizada por *hackers* e profissionais de tecnologia da informação em geral.

O computador utilizado para realizar as simulações foi conectado à Internet através de uma rede externa ao IFRJ. Dessa maneira, pôde-se simular como seriam as consequências de ataques reais provenientes de fora da rede da instituição. Para alguns ataques, foi necessária a utilização de técnicas para burlar o *firewall* do campus, já que o mesmo está configurado para bloquear grande parte das conexões maliciosas mais comuns através de filtragem de pacotes. Em momento algum o *firewall* foi desabilitado ou reconfigurado durante as simulações de ataques. Por questões de sigilo e segurança, as técnicas utilizadas para realização dos testes de ataque não serão mostradas neste trabalho. A Figura 4.8 representa como foi feita a conexão desse computador utilizado para os testes, onde é executado o Kali Linux.

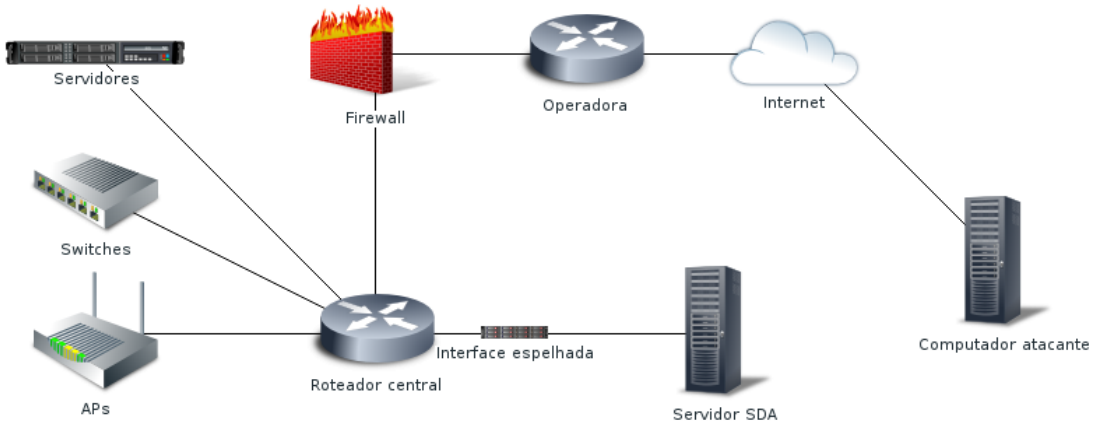


Figura 4.8: Conexão do computador utilizado para as simulações da ataque

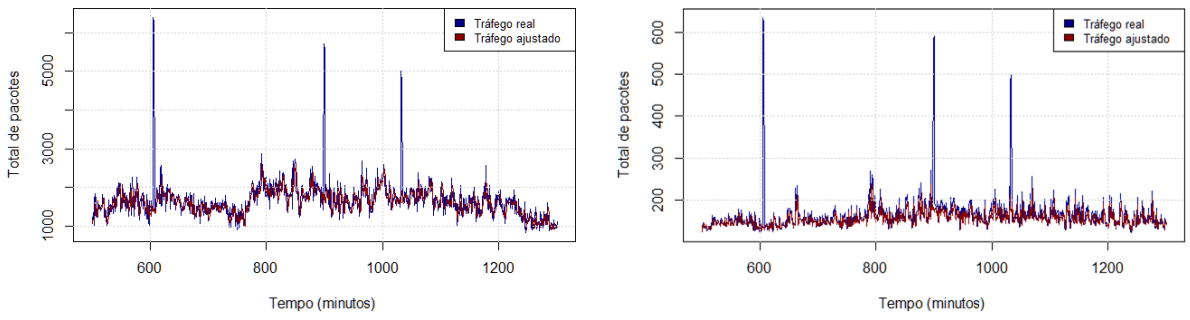
As primeiras simulações de ataques foram realizadas nos dias 04 e 05 de junho de 2018. Neste período não foram observadas quaisquer anomalias reais no tráfego, e o uso dos serviços de rede, a disponibilidade e o desempenho dos recursos foi considerado como normal. Assim sendo, pode-se considerar que todas as anormalidades observadas nos tráfegos de pacotes nesse período foram causadas especificamente pelos ataques induzidos.

Como o desenvolvimento do módulo de detecção de anomalias do SDA já havia sido concluído, o sistema estava ativo e funcionando conforme o esperado. Assim sendo, foi possível detectar as anomalias induzidas e identificar os ataques, com envio de alertas de teste por e-mail para o gerente da rede.

A seguir são apresentados os resultados das simulações de cada um dos tipos de ataque de negação de serviço apresentados neste trabalho, descritos nas Seções 2.1.5 e 3.3.3. Para cada simulação, foram construídos gráficos relativos às métricas relacionadas ao ataque em questão, conforme disposto na Tabela 3.1. O número de ataques gerados em cada simulação está relacionado à quantidade de métricas referentes a cada tipo de ataque, de acordo com a tabela. Dessa forma, para os tipos de ataque que possuem até três métricas relacionadas, foram induzidos dois ou três ataques, e para aqueles com quatro métricas relacionadas (no caso, apenas o *HTTP Floods*), foram induzidos quatro ataques.

4.3.2 *Fraggle Attack*, *Smurf Attack* e *DNS Amplification*

Primeiramente foram gerados três ataques do tipo *Fraggle Attack*. Conforme era esperado, foram detectadas anomalias no tráfego de pacotes referentes às métricas *UDP* e *ICMP*, nos instantes de tempo 605, 899 e 1032, e o ataque foi identificado e sinalizado com sucesso nos três testes. Os gráficos relativos ao tráfego de pacotes das duas métricas durante a simulação de ataques são mostrados na Figura 4.9.



(a) Métrica *UDP* (b) Métrica *ICMP*

Figura 4.9: Simulação de ataques do tipo *Fraggle Attack*

De acordo com os gráficos da Figura 4.9, observa-se que o alto número de pacotes UDP trafegados é proveniente do *host* atacante. O tráfego anômalo de pacotes ICMP se deve à resposta do *host* vítima, que tenta responder às requisições do atacante com pacotes desse tipo.

Posteriormente, também foi obtido sucesso com a indução de três ataques do tipo *DNS Amplification*. Foram detectadas anomalias nas métricas *DNS*, *UDP* e *ICMP* nos três instantes de tempo em que os ataques ocorreram, conforme esperado de acordo com a Tabela 3.1. Como cada requisição DNS é realizada com o protocolo UDP na camada de transporte, a quantidade anômala de pacotes da métrica *DNS* é a mesma dos pacotes UDP. Assim como no *Fraggle Attack*, o host vítima responde às requisições gerando um tráfego intenso de pacotes ICMP. Adicionalmente, nota-se que a proporção aproximada de pacotes ICMP enviados como resposta é de um para cada dez pacotes UDP recebidos, tanto para ataques *DNS Amplification* quanto para *Fraggle Attack*.

Também foram detectadas anomalias no tráfego de pacotes ICMP durante a simulação de ataques do tipo *Smurf Attack*. Esse ataque foi facilmente identificado, pois somente a métrica *ICMP* é afetada.

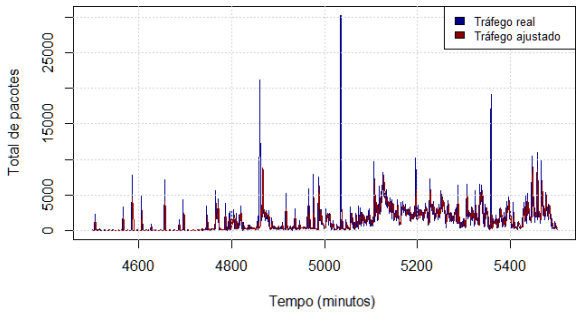
É interessante notar que o SDA deve ser capaz de diferenciar a ocorrência entre ataques do tipo *Smurf Attack*, *Fraggle Attack* e *DNS Amplification*. Como os três ataques geram anomalias na métrica *ICMP*, o sistema foi programado de forma a verificar a ocorrência simultânea de anomalias também nas métricas *UDP* e *DNS*, para que os ataques possam ser identificados corretamente. Entretanto, em casos de ocorrência de um *Smurf Attack* ao mesmo tempo de um *Fraggle Attack*, por exemplo, somente o *Fraggle Attack* seria identificado. Analogamente, esses dois tipos de ataque poderiam também não ser identificados caso ocorressem ao mesmo tempo em que um *DNSAmplification*. Para contornar esse problema, ao enviar um alerta de ocorrência de um *Fraggle Attack*, o sistema também sinaliza a possibilidade de ocorrência simultânea de um *Smurf Attack*. Da mesma forma, também são sinalizadas possibilidades de ocorrência de *Smurf Attack* e *Fraggle Attack* durante o alerta de ocorrência de um *DNS Amplification*.

4.3.3 *HTTP Floods*, *TCP SYN Flood* e *TCP Reset Attack*

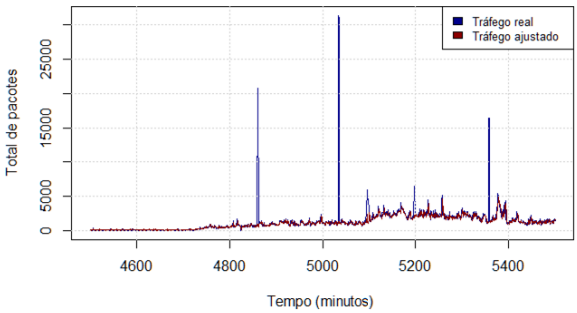
Resultados interessantes foram obtidos durante a simulação de ataques *HTTP Floods*. Foram observadas anomalias nas métricas *HTTP*, *TCP_SYN*, *TCP_SYN_ACK* e *TCP_RST_ACK*. Nessa última métrica era não esperado que ocorressem anomalias, de acordo com a Tabela 3.1. Pode-se concluir que isso se deve ao fato de o *host* vítima rejeitar as tentativas de conexão mal sucedidas enviando pacotes TCP com as *flags RST* e *ACK* ativadas. Para esse ataque, era esperado que também fossem detectadas anomalias na série temporal da métrica *TCP_ACK*. Contudo, o tráfego gerado não foi suficiente para influenciar no tráfego dos pacotes dessa métrica, visto que o número de

pacotes normalmente transmitidos é muito maior. Recorre-se à Figura 4.1 para confirmar esse fato.

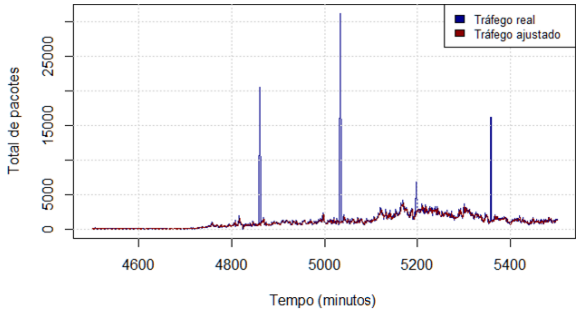
Os resultados da indução de quatro ataques do tipo *HTTP Floods* são apresentados na Figura 4.10. Os ataques ocorreram nos instante de tempo 4860, 5034, 5196 e 5357.



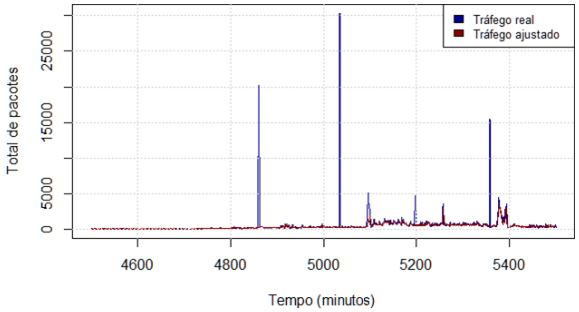
(a) Métrica *HTTP*



(b) Métrica *TCP_SYN*



(c) Métrica *TCP_SYN_ACK*



(d) Métrica *TCP_RST_ACK*

Figura 4.10: Simulação de ataques *HTTP Floods*

As anomalias foram detectadas com sucesso em todas as séries temporais no primeiro, no segundo e no quarto ataque. Entretanto, como o terceiro ataque foi menos intenso (gerou uma quantidade menor de pacotes), o SDA não detectou anomalia na métrica *HTTP*. Em vez disso, foi identificado erroneamente um ataque do tipo *TCP SYN Flood*, que está relacionado justamente com as métricas *TCP_SYN*, *TCP_SYN_ACK* e *TCP_RST_ACK*.

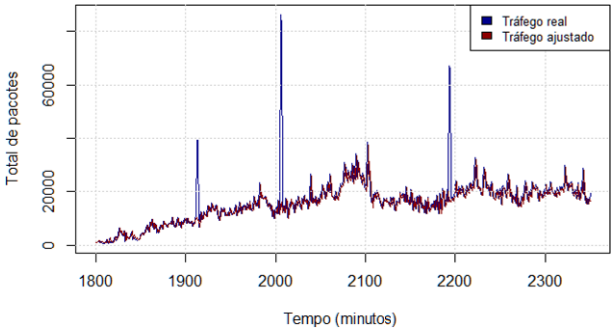
Além disso, o sistema sinalizou uma anomalia também no instante 5098 para as métricas *TCP_SYN* e *TCP_RST_ACK*. Essas sinalizações tratam-se de falsos positivos, causados por tráfegos classificados incorretamente como anômalos.

Posteriormente foram induzidos ataques *TCP SYN Flood*, em três instantes de tempo. Todos os ataques desse tipo foram identificados e sinalizados corretamente, através da detecção de anomalias nas séries temporais das três métricas relacionadas ao ataque.

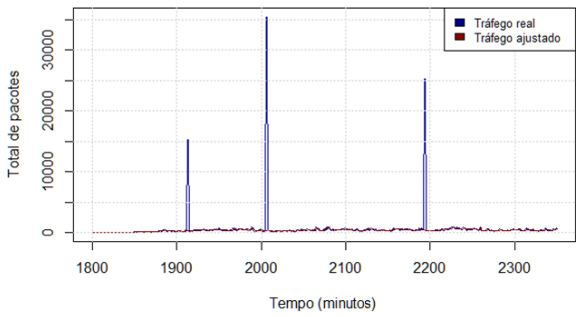
Ao simular ataques do tipo *TCP Reset Attack*, as detecções também foram realizadas com sucesso, sendo observadas anomalias somente nos pacotes da métrica TCP_RST, conforme esperado.

4.3.4 *TCP PSH+ACK Flood*, *TCP FIN+ACK Flood* e *Spoofed Session Attack*

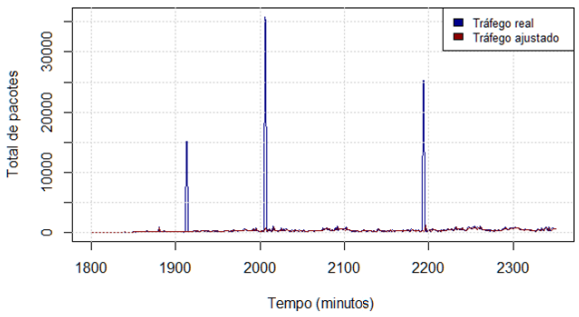
Ao simular três ataques do tipo *TCP PSH+ACK Flood*, foram detectas anomalias no tráfego de pacotes das métricas relacionadas e os ataques foram sinalizados corretamente. Conforme mostrado na Figura 4.11, foram geradas anomalias nos instantes de tempo 1913, 2006 e 2193.



(a) Métrica TCP_PSH_ACK



(b) Métrica TCP_RST



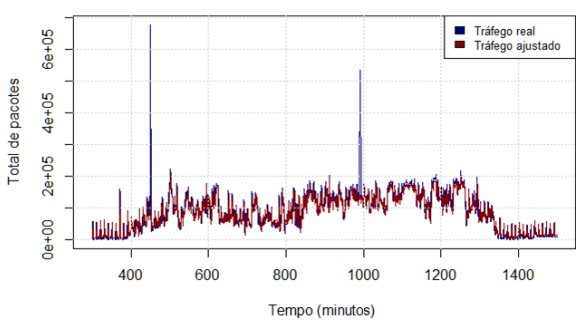
(c) Métrica TCP_RST_ACK

Figura 4.11: Simulação de ataques *TCP PSH+ACK Flood*

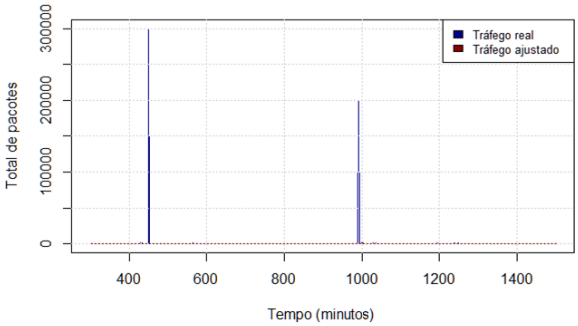
Além das métricas TCP_PSH_ACK e TCP_RST, previstas na Tabela 3.1, também foram observadas anomalias no tráfego de pacotes da métrica TCP_RST_ACK. Estudando os efeitos do ataque em mais detalhes, foi observado que esse tráfego foi gerado pelo próprio *host* atacante, e não pela vítima (esta responde aos pacotes TCP $PSH+ACK$ com pacotes TCP RST). Esse fato ocorre devido a particularidades da ferramenta utilizada para realizar os ataques.

Isso também ocorreu durante a simulação de ataques *TCP FIN+ACK Flood* e *Spoofed Session Attack*, apesar de todos os ataques terem sido sinalizados corretamente. Em ataques do tipo *TCP FIN+ACK Flood*, eram esperadas anomalias somente na série temporal da métrica TCP_FIN_ACK, porém também foi gerado um tráfego anômalo de pacotes TCP_RST_ACK pelo *host* atacante. Já para o *Spoofed Session Attack*, esperavam-se anomalias somente nas métricas TCP_ACK (pacotes gerados pelo atacante) e TCP_RST (pacotes gerados pela vítima). Nesse tipo de ataque, o atacante também gerou um tráfego anormal de pacotes TCP $RST + ACK$.

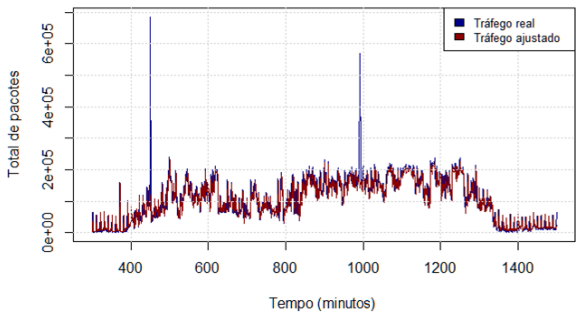
Outro fato interessante foi que os ataques do tipo *Spoofed Session Attack* também geraram anomalias na série temporal da métrica $TOTAL\_PACOTES$. Como o número de pacotes necessários para causar uma anomalia no tráfego de pacotes TCP ACK é muito grande, esse número foi suficiente para influenciar também no tráfego total. A Figura 4.12 contém o resultado da simulação e identificação de dois ataques desse tipo, nos instantes de tempo 450 e 991.



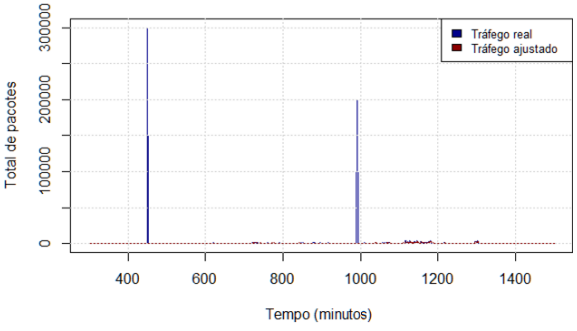
(a) Métrica TCP_ACK



(b) Métrica TCP_RST



(c) Métrica $TOTAL\_PACOTES$



(d) Métrica TCP_RST_ACK

Figura 4.12: Simulação de ataques *Spoofed Session Attack*

4.3.5 Resultados e Discussões Adicionais

Nos dois dias posteriores aos primeiros testes de ataques, foram feitas novas simulações, através da indução dos mesmos ataques, mas de maneira individual e simultânea. Entende-se por *falsos positivos* todas as sinalizações equivocadas de ataques ou anomalias, em instantes em que na verdade não ocorreram eventos anômalos no tráfego, porém o sistema acusou a ocorrência. *Falsos negativos* são ocasiões em que o sistema não detectou uma anomalia ou um ataque que estava realmente ocorrendo. A Figura 4.13 contém um gráfico que mostra o total dos ataques induzidos que foram identificados com sucesso, dos que foram identificados incorretamente e daqueles que não foram identificados (falsos negativos). Durante esse período, foram gerados 4 falsos positivos de sinalizações de anomalias no tráfego.

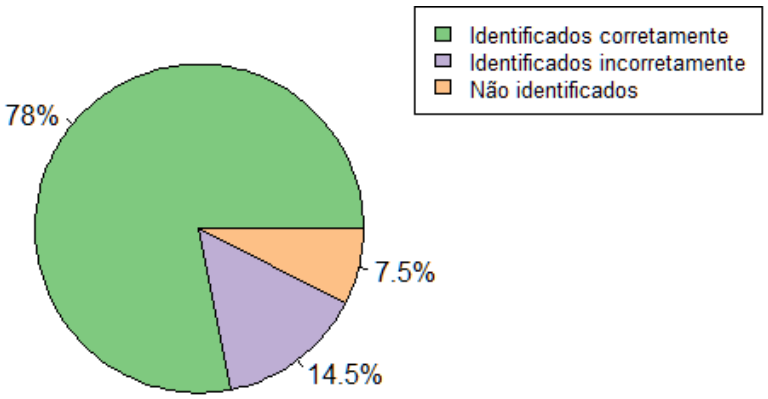


Figura 4.13: Identificação de ataques de negação de serviço induzidos

É muito importante ressaltar que apesar de algumas inconsistências como a ocorrência de falsos positivos e falsos negativos para algumas séries temporais, além da necessidade de revisão da associação de métricas para alguns ataques na Tabela 3.1, para reduzir o número de identificações incorretas, as simulações realizadas obtiveram sucesso na identificação de todos os tipos de ataque de negação de serviço abordados neste trabalho. O método proposto se mostrou bastante eficaz ao ser testado no ambiente real de estudo.

Além de identificar ataques de negação de serviço, o SDA também é capaz de detectar anomalias mais genéricas, que não estejam previamente relacionadas a nenhum ataque específico. A identificação de anomalias na série temporal da métrica $OUTROS\_PACOTES$, por exemplo, pode sinalizar a ocorrência de algum tipo de anormalidade que não havia sido prevista.

Uma anomalia no tráfego de um determinado tipo de pacote pode ocorrer por outros motivos além de um ataque malicioso. Exemplos comuns são a ocorrência de um número grande de requisições a um servidor em uma ocasião específica e a transmissão em larga escala de vídeos por *streaming*. A ocorrência de muitos falsos positivos pode se tornar um inconveniente ao longo do tempo, caso sejam enviados muitos alertas incorretos para o gerente da rede. Por esse motivo, propõe-se continuar estudando e observando o comportamento do tráfego de rede e os resultados fornecidos pelo Sistema de Detecção de Anomalias, e implementar uma melhoria contínua da metodologia, buscando formas de diminuir o número de falsos positivos e consolidar o sistema como uma ferramenta cada vez mais poderosa capaz de detectar os ataques de negação de serviço que ocorram na rede, auxiliando os profissionais da instituição a identificar vulnerabilidades nos sistemas e lidar proativamente com questões relacionadas à segurança da informação.

4.4 Contribuições

O campus Volta Redonda do IFRJ conta com uma poderosa ferramenta de monitoramento de rede e segue rígidos padrões institucionais de segurança da informação e de privacidade. Mas por ser uma instituição pública, o IFRJ muitas vezes enfrenta condições precárias em relação a investimentos, e essa realidade afeta também o cenário tecnológico. Assim sendo, torna-se necessário a busca por soluções e tecnologias de baixo custo para manter e melhorar a infraestrutura e os serviços de TI e telecomunicações. Destaca-se então uma das grandes vantagens da implementação do método proposto nesta dissertação, que é possuir um baixo custo operacional. Vale destacar também a utilização de *softwares* e linguagens de programação livres no desenvolvimento do método.

Assim sendo, a principal contribuição do presente trabalho de dissertação é prover um mecanismo de detecção preventiva de anomalias em uma rede real. Essa detecção se dá de forma dinâmica e automatizada, através de alertas ao gerente de rede sobre a ocorrência de eventos suspeitos, servindo assim como ferramenta auxiliar para tomada de decisões. Além de minimizar o custo operacional, a metodologia aqui proposta utiliza técnicas conhecidas de previsão por séries temporais, que são largamente estudadas e possuem uma complexidade relativamente simples de implementação computacional.

Dentre as atribuições mais comuns de um gerente de rede relativas à segurança da informação, pode-se destacar a criação e atualização de regras e políticas de segurança; a garantia da integridade, da confidencialidade e da disponibilidade das informações; e a

verificação de ocorrências de infrações à segurança. Uma vez que o SDA emite um alerta através de e-mail para o gerente de rede quando um ataque é identificado, caberá então a esse profissional analisar a situação e tomar as devidas providências que lhe competem. Assim sendo, o gerente passa a contar com uma importante ferramenta capaz de auxiliá-lo no estudo e na averiguação das vulnerabilidades e falhas de segurança que permitiram que tais eventos maliciosos ocorressem, a fim de impedir ocorrências futuras da mesma ameaça.

Na Seção 3.3.1, foi explicado que o módulo de captura de tráfego do SDA grava as informações referentes ao tráfego de pacotes em arquivos temporários do TShark. Um recurso interessante adicionado ao sistema é a disponibilização do conteúdo desses arquivos para o gerente de rede em caso de detecção de anomalias, para que ele possa analisar mais a fundo o comportamento anormal que ocorreu no tráfego, além de identificar os endereços IP de origem e destino dos *hosts* envolvidos, por exemplo.

Em períodos anteriores aos testes expostos na Seção 4.3, o sistema sinalizou a detecção de anomalias muito discrepantes no tráfego durante a madrugada, que evidenciam a possibilidade de que tenham acontecido ataques maliciosos à rede do campus. Esse fato é muito importante de se destacar, pois além de ter mostrado a capacidade do SDA de detectar ataques reais, ressalta a necessidade de elaboração de novas políticas de segurança para a rede da instituição.

De forma resumida, a Tabela 4.4 apresenta uma comparação dos principais trabalhos relacionados descritos na Seção 2.3 com o presente trabalho de dissertação, apresentando as características mais importantes.

Trabalho	Objetivo principal	Modelos de previsão	Cenário de estudo	Linguagens e tecnologias utilizadas	Análise do tráfego	Granularidade	Análise do tempo computacional	Intervalo de confiança	Métricas
[42]	Previsões de longo prazo	ARIMA	Rede real	Não informado	Pacotes	1 dia e 1 semana	Não	Não informado	Não aplicável
[3]	Encontrar o modelo mais adequado para previsão	ARIMA	Rede real	Não informado	Pacotes	1 hora e 1 dia	Não	Não informado	Não aplicável
[10]	Previsões de longo prazo	ARIMA e SARIMA	Rede real e dados simulados	Apenas livres	Fluxos	1 dia e 1 mês	Não	95%	Não aplicável
[7]	Análise de modelos híbridos	Diversos	Dados reais disponíveis publicamente	Não informado	Pacotes	Diversas	Sim	Diversos	Não aplicável
[43]	Previsões de curto prazo	Diversos	Redes reais	Não informado	Pacotes	15 minutos e 1 hora	Não	Não informado	Não aplicável
[2]	Caracterizar o padrão de comportamento do tráfego	Diversos	Redes reais	Não informado	Pacotes	Diversas	Não	Não informado	Próprias
[1]	Detectar perda de pacotes em tempo real	ARIMA	Rede real	Não informado	Fluxos	10 segundos	Não	Diversos	Próprias
[6]	Revisão de técnicas propostas em outros trabalhos	Diversos	Diversos	Não informado	Fluxos e pacotes	Diversas	Não	Não informado	Não aplicável
[44, 45]	Detecção de 3 tipos de ataque de negação de serviço	ARIMA	Rede real	Livres e proprietárias	Pacotes	15 segundos	Não	Não informado	Próprias
[46]	Detecção de anomalias	Holt-Winters	Rede real	Apenas livres	Bits/s	5 minutos	Não	95%	Próprias
[25]	Detecção de TCP SYN Flood	Limite adaptativo e CUSUM	Rede real	Apenas livres	Pacotes	Não informada	Não	Não informado	Baseadas em outros trabalhos
[23]	Detecção de 4 tipos de ataque de negação de serviço e 3 tipos de ataque de escaneamento de porta	Holt-Winters	Rede real	Livres e proprietárias	Fluxos	5 minutos	Não	95%	Baseadas em outros trabalhos
[48]	Detecção de 3 tipos de ataque de negação de serviço e 1 tipo de ataque de escaneamento de porta	Holt-Winters	Rede real	Livres e proprietárias	Fluxos	Não informada	Não	Não informado	Próprias
[4]	Detecção ataques de negação de serviço e escaneamento de porta	Híbridos	Rede real	Não informado	Fluxos e pacotes	15 minutos	Não	Não informado	Próprias
Este trabalho	Detecção de 9 tipos de ataque de negação de serviço e outras anomalias	ARIMA e SARIMA	Rede real	Apenas livres	Pacotes	1 minuto	Sim	95%	Próprias

Tabela 4.4: Comparação das características dos principais trabalhos relacionados com o presente trabalho

Capítulo 5

Conclusões e Trabalhos Futuros

Este capítulo apresenta as conclusões obtidas com a realização deste trabalho, além de sugestões de trabalhos futuros.

5.1 Conclusões

Ataques de negação de serviço são ameaças relacionadas à segurança da informação que podem causar grandes problemas no desempenho e disponibilidade de diversos recursos computacionais em uma rede de computadores, podendo causar grandes prejuízos e lesar a qualidade de serviços essenciais para o funcionamento de uma organização. A detecção antecipada dessas ameaças se torna cada vez mais necessária com a constante evolução tecnológica e o surgimento de novas vulnerabilidades.

Este trabalho tem como principal objetivo desenvolver uma solução computacional para detecção de ataques de negação de serviço, através da modelagem por séries temporais do tráfego de pacotes da rede do IFRJ - campus Volta Redonda. Detectar anomalias no tráfego e identificar ataques em tempo real eficientemente é um grande desafio relacionado à segurança da informação. A realização deste trabalho foi motivada pelo fato de a rede da instituição carecer de mecanismos eficientes capazes de lidar com ameaças à segurança de maneira preventiva, servindo de suporte ao gerente de rede para a análise de fragilidades que existam na rede e nos sistemas.

Os resultados dos testes de utilização do sistema na prática mostram que o tráfego de rede do campus pode ser representado com precisão ao ser modelado como séries temporais. Os modelos de previsão utilizados para cada série temporal são baseados em ARIMA e SARIMA, e a seleção desses modelos é feita de forma automatizada através da

função *auto.arima* do R, principal linguagem de programação utilizada no trabalho. Os modelos selecionados foram avaliados e os ajustes obtidos mostraram-se adequados para as séries, podendo ser utilizados para realizar previsões confiáveis.

As métricas definidas neste trabalho têm a finalidade de classificar e dividir o tráfego em grupos, de acordo com os tipos de pacotes que são transmitidos na rede. Analisando cada métrica como uma série temporal, é possível detectar ataques específicos de forma mais precisa, baseado no tráfego anômalo de pacotes.

O método proposto foi testado e validado através da análise dos dados capturados referentes ao tráfego na rede estudada, e de simulações de ataques de negação de serviço. Através dessas simulações, pode-se comprovar que o sistema desenvolvido é capaz de sinalizar anomalias no tráfego e identificar todos os ataques abordados neste trabalho em tempo real, de forma eficiente e satisfatória. Assim, conclui-se que o método pode ser utilizado na rede de computadores do campus para identificar ameaças reais.

A análise dos resultados mostrou também a importância da utilização de métricas cujo número de pacotes trafegado é muito menor em relação às outras. Com essas métricas, os ataques geram anomalias mais evidentes, sendo mais fáceis de serem detectadas. Além disso, a identificação simultânea de anomalias em métricas diferentes é capaz de evidenciar ataques distintos.

Apesar dos resultados significativos obtidos com os testes e simulações, ainda são necessárias correções e melhorias no método proposto, no que tange a captura de tráfego e identificação de ataques, incluindo correções na Tabela 3.1, que associa cada ataque com as métricas propostas, e também o estudo de soluções e evolução contínua que visam diminuir o número de falsos positivos e falsos negativos, melhorando a capacidade do sistema de sinalizar anomalias no tráfego de forma cada vez mais precisa.

Ademais, destaca-se a possibilidade da aplicação da metodologia proposta neste trabalho em outros campus do IFRJ e até mesmo em outras instituições, que possuam cenários similares à rede aqui estudada.

5.2 Trabalhos Futuros

A realização deste trabalho de dissertação motiva diversas possibilidades de trabalhos futuros. Como uma das propostas, sugere-se a modelagem e previsão do tráfego da rede com outros modelos de séries temporais diferentes de ARIMA e SARIMA, como o modelo de Holt-Winters, e também a utilização de técnicas de redes neurais artificiais, *machine learning* (aprendizado de máquina), análise de sinais e até mesmo métodos híbridos, a fim de comparar os resultados obtidos em relação à identificação de ataques.

Além da utilização de outras técnicas, com o objetivo de obter previsões mais precisas, com valores menores para o MAPE, pretende-se fazer análises e simulações com granularidades diferentes de 1 minuto e com janelas de amostragem diferentes de 2 semanas. Dessa forma, espera-se reduzir o número de falsos negativos.

A definição de mais de um limite superior também é uma interessante melhoria a ser implementada, com o objetivo de diferenciar a intensidade dos ataques através da emissão de alertas em diferentes níveis, além de diminuir o número de ocorrências de falsos positivos. Outro recurso interessante também a ser acrescentado é a análise de limites inferiores, a fim de observar possíveis reduções no número de alguns tipos de pacote durante a ocorrências de ataques.

A aplicação de outras técnicas estatísticas pode ser incorporada no procedimento de detecção de anomalias, com o objetivo de reduzir o número de falsos positivos durante períodos em que ocorram picos no tráfego da rede que não sejam causados por ataques maliciosos, mas sim pelo uso mais intenso dos recursos computacionais em ocasiões específicas.

Mesmo com resultados favoráveis, as simulações de ataques mostraram a necessidade de melhorias na associação entre métricas e ataques, principalmente para aqueles cuja identificação depende da detecção de anomalias no número de pacotes enviados como resposta pelo *host* vítima. Em caso de indisponibilidade desse *host*, alguns ataques podem não ser sinalizados corretamente. Uma das sugestões é comparar os resultados obtidos neste trabalho com resultados decorrentes da substituição de algumas métricas por outras utilizadas em alguns trabalhos, como [4, 23]. A adição de métricas que considerem o tamanho dos pacotes também é uma interessante melhoria a ser adicionada. Outro estudo interessante que pode ser feito é o da possibilidade de redução do número de métricas utilizadas para identificar os ataques, baseando-se na análise da correlação linear entre as séries temporais, assim como foi feito no trabalho [9].

Vale destacar também o estudo de outras formas de lidar com as séries temporais no momento em que as anomalias são detectadas. Ao descartar os valores anômalos para fazer as previsões, sugere-se considerar, por exemplo, a repetição do último valor não anômalo da série, ao invés de utilizar a própria previsão feita para o instante atual para compor a série temporal e prever o próximo valor. Essa abordagem pode ter um impacto positivo na redução do erro nos ajustes feitos com os modelos de previsão.

Dadas as pré-condições de monitoramento do ambiente real de estudo, foi possível considerar que o tráfego observado no período de análise dos resultados está livre de atividades anormais e maliciosas, sendo caracterizado como um tráfego comum e com comportamento padrão. Ainda assim, destaca-se também a possibilidade de realização de testes de ataque em um ambiente simulado e/ou controlado, com o objetivo de analisar o impacto das anomalias e o comportamento das métricas.

Por fim, outra sugestão é expandir o escopo do sistema, adicionando mecanismos que sejam capazes de identificar outros tipos de ataques de negação de serviço e também ataques de escaneamento de porta, que são ameaças muito estudadas e utilizadas por *hackers* para encontrar vulnerabilidades em uma rede. Esses ataques também podem ser detectados através da ocorrência de anomalias no tráfego.

Referências

[1] Vafeiadis, T., Papanikolaou, A., Ilioudis, C., Charchalakis, S. Real-time network data analysis using time series models. *Elsevier Simulation Modelling Practice and Theory 29* (2012), 173–180. http://dx.doi.org/10.1016/j.simpat.2012.07.002.

[2] Santos, A. C. F., da Silva, J. D. S., de Sá Silva, L., da Costa Sene, M. P. Network traffic characterization based on time series analysis and computational intelligence. *Journal of Computational Interdisciplinary Sciences 2(3)* (2011), 197–205. doi: 10.6062/jcis.2011.02.03.0046.

[3] Jung, S., Kim, C., Chung, Y. A prediction method of network traffic using time series models. *Computational Science and Its Applications-ICCSA* (2006), 234–243. http://dx.doi.org/10.1007/11751595_26.

[4] Hong, W., Zhenghu, G., Qing, G., Baosheng, W. Detection network anomalies based on packet and flow analysis. *Seventh International Conference on Networking. School of Computer, National University of Defense Technology, Changsha 410073, China* (Abril de 2008), 497–502. DOI 10.1109/ICN.2008.83.

[5] Schilling, A. A framework for secure IT operations in an uncertain and changing environment. *Elsevier Computers and Operations Research 85* (2017), 139–153. http://dx.doi.org/10.1016/j.cor.2017.04.008.

[6] Joshi, M. R., Hadi, T. H. A review of network traffic analysis and prediction techniques. *School of Computer Sciences, North Maharashtra University, Jalgaon (M.S), India* (2015).

[7] Katris, C., Daskalaki, S. Comparing forecasting approaches for internet traffic. *Elsevier Expert Systems with Applications 42* (2015), 8172–8183. http://doi.org/10.1016/j.eswa.2015.06.029.

[8] Hinich, M. J., Molyneux, R. E. Predicting information flows in network traffic. *Journal of the American Society for Information Science and Technology 54(2)* (2003), 161–168. http://dx.doi.org/10.1002/asi.10176.

[9] Spagnol, R. L. Modelagem de redes de computadores por métodos estatísticos. Dissertação de Mestrado, Universidade Estadual de Campinas - UNICAMP, Brasil, Dezembro de 2011.

[10] Macedo, E. L. C. Previsão de tráfego em enlaces de redes utilizando séries temporais. Dissertação de Mestrado, COPPE/UFRJ, Brasil, Setembro de 2015.

[11] IFRJ. Instituto Federal de Educação, Ciência e Tecnologia do Rio de Janeiro. Junho de 2018. Disponível em: http://portal.ifrj.edu.br/.

[12] Tanenbaum, A. S., Wetherall, D. J. *Computer Networks*, quinta ed. Prentice Hall. Boston, Massachusetts, USA, 2011. ISBN: 9780132126953.

[13] Senac. *Guia Internet de Conectividade*, 11ª ed. Senac São Paulo, 2000. Cyclades Brasil.

[14] Kurose, J. F., Ross, K. W. *Computer Networking: A Top-Down Approach*, quinta ed. Addison-Wesley Publishing Company, USA, 2009. ISBN: 0136079679, 9780136079675.

[15] RFC. Request for Comments. Junho de 2018. Disponível em: http://www.ietf.org/rfc.html.

[16] Northcutt, S. *Network Intrusion Detection: An Analyst's Handbook*. New Riders Publishing. Thousand Oaks, CA, USA, Setembro de 2003. ISBN: 0735708681.

[17] Paranet Solutions. Tip of the Week: A TCP flags and handshakes walk-through. Junho de 2018. Disponível em: http://www.paranet.com/blog/bid/147655/Tip-of-the-Week-A-TCP-flags-and-handshakes-walk-through.

[18] Oetiker, T. MRTG: The Multi Router Traffic Grapher. *Proceedings of the 12th Conference on Systems Administration* (Dezembro de 1998), 141–148. http://www.usenix.org/publications/library/proceedings/lisa98/oetiker.html.

[19] Zabbix. The Enterprise-Class Open Source Network Monitoring Solution. Junho de 2018. Disponível em: http://www.zabbix.com/.

[20] Wireshark. Go Deep. Junho de 2018. Disponível em: http://www.wireshark.org/.

[21] TCPDUMP. Manpage of TCPDUMP. Junho de 2018. Disponível em: http://www.tcpdump.org/.

[22] Marchette, D. J. *Computer Intrusion Detection and Network Monitoring: A Statistical Viewpoint, v. 1, Statistics for Engineering and Information Science*, primeira ed. Springer New York. New York, USA, 2001. ISBN: 9781441929372, 9781475734584. doi: 10.1007/978-1-4757-3458-4.

[23] da Silva, V. L. P. Identicação de anomalias em fluxos de rede utilizando previsões em séries temporais pelo método de Holt-Winters. Dissertação de Mestrado, COPPE/UFRJ, Brasil, Setembro de 2015.

[24] JavaPipe: Cloud Hosting & DDoS Protection Expert. (Distributed) Denial of Service Attack: Definition & Prevention. Junho de 2018. Disponível em: http://javapipe.com/ddos/blog/denial-of-service-attack/.

[25] Salunkhe, H. S., Jadhav, S., Bhosale, V. Analysis and review of TCP SYN Flood Attack on network with its detection and performance metrics. *International Journal of Engineering Research & Technology (IJERT) 6(01)* (2017), 250–256. ISSN: 2278-0181.

[26] CGI.br. *Cartilha de Segurança para Internet*. CERT.br - Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil, Outubro de 2006. Versão 3.1. http://cartilha.cert.br/.

[27] Bowerman, B. L., O'Connel, R. *Forecasting and Time Series: an Applied Approach.* Belmont. Duxbury Press, 1993.

[28] Chatfield, C. *Time-Series Forecasting.* CHAPMAN & HALL/CRC, 2000. ISBN: 1584880635.

[29] Morettin, P. A., Toloi, C. M. C. *Modelos para Previsão de Séries Temporais*, segunda ed. Atual Editora, São Paulo, Brasil, 2004.

[30] Abraham, B., Ledolter, J. *Statistical methods for forecasting.* Wiley, 1983. ISBN: 9780471867647.

[31] R: The R Project for Statistical Computing. Junho de 2018. Disponível em: http://www.r-project.org/.

[32] Kleiber, C., Zeileis, A. *Applied Econometrics with R.* Springer Science+Business Media, 2008. DOI: 10.1007/978-0-387-77318-6. ISBN: 9780387773162, 9780387773186.

[33] Monolito Nimbus. O conhecimento está em toda parte. Junho de 2018. Disponível em: https://www.monolitonimbus.com.br/.

[34] de Jesus, J. C. Modelagem do crescimento de glioma por séries temporais em resposta à radioterapia. Dissertação de Mestrado, Universidade Federal Fluminense - UFF, Brasil, Setembro de 2014.

[35] Makridakis, S., Wheelwright, S. C., Hyndman, R. J. *Forecasting: Methods and Applications*, terceira ed. Springer New York. New York, USA, 1998.

[36] Box, G. E. P., Jenkins, G. M. *Time series analysis forecasting and control.* Holden-Day. San Francisco, USA, 1976.

[37] Bertolo, L. A. *Técnicas de Previsão de Box-Jenkins no Excel.* IMES - Catanduva. http://www.bertolo.pro.br/.

[38] Akaike, H. A new look at the statistical model identification. *IEEE Transactions on Automatic Control 19* (1974). ISSN: 0018-9286. doi: 10.1109/TAC.1974.1100705.

[39] Morettin, P. A., Bussab, W. O. *Estatística Básica*, quinta ed. Saraiva, São Paulo, Brasil, 2005.

[40] Wei, W. W. *Time Series Analysis - Univariate and Multivariate Methods*, segunda ed. Departament of Statistcs of The Fox School of Business and Management, 2006.

[41] Bozdogan, H. Model selection and Akaike's information criterion (AIC): The general theory and its analytical extensions. *Psychometrica 52* (1987), 345–370.

[42] Groschwitz, N. K., Polyzos, C. A time series model of long-term NSFNET backbone traffic. *IEEE International Conference on Communications 3* (1994), 1400–1404.

[43] Klevecka, I. Forecasting network traffic: A comparison of neural networks and linear models. *Proceedings of the 9th International Conference "Reliability and Statistics in Transportation and Communication"* (Outubro de 2009), 170–177. ISBN 9789984818214.

[44] Lunardi, R. C. *Um Analisador De Intrusões Baseado Em Séries Temporais*. Universidade Federal de Santa Maria, Brasil, 2008. Trabalho de Graduação nº 255.

[45] Lunardi, R. C., Dalmazo, B. L., do Amaral, E. M. H., Nunes, R. C. DIBSet: um detector de intrusão por anomalias baseado em séries temporais. *VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais* (2008), 311–230.

[46] Brutlag, J. D. Aberrant behavior detection in time series for network monitoring. *Proceedings of the 14th USENIX Conference on System Administration* (2000), 139–146. http://dl.acm.org/citation.cfm?id=1045502.1045530.

[47] de Abreu Silva, C. Proposta de implementação de uma ferramenta para gerência em redes baseada numa nova metodologia usando análise de tráfego em backbones ip. Dissertação de Mestrado, COPPE/UFRJ, Brasil, 2006.

[48] Nguyen, H. A., Nguyen, T. T. V., Kim, D. I. , et.al. Network traffic anomalies detection and identification with flow monitoring. *Wireless and Optical Communications Networks. 5th IFIP International Conference* (Maio de 2008), 1–5. doi: 10.1109/WOCN.2008.4542524.

[49] RNP. Rede Nacional de Ensino e Pesquisa. Junho de 2018. Disponível em: http://www.rnp.br/.

[50] CentOS Project. Junho de 2018. Disponível em: http://www.centos.org/.

[51] MySQL. The world's most popular open source database. Junho de 2018. Disponível em: http://www.mysql.com/.

[52] Portal Action. Junho de 2018. Disponível em: https://www.portalaction.com.br/.

[53] Camilo, E. V., Maestre, M. R., Mangueira, R. A. F., de Medeiros, E. S., Villegas, C. Utilizando a função auto.arima em modelos de séries temporais. http://www.rbras.org.br/rbras58/sites/default/files/submissoes/Autoarima_Erasnilson.pdf.

[54] Kalutarage, H. K., Shaikh, S. A., Wickramasinghe, I. P., Zhou, Q., James, A. E. Detecting stealthy attacks: Efficient monitoring of suspicious activities on computer networks. *Elsevier Computers and Electrical Engineering 47* (2015), 327–344. http://dx.doi.org/10.1016/j.compeleceng.2015.07.007.

[55] UML. Unified Modeling Language. Junho de 2018. Disponível em: http://www.uml.org/.

[56] Klevecka, I. Short-term traffic forecasting with neural networks. *Transport and Telecommunication Institute, Lomonosova 1, Riga, LV-1019, Latvia 12(2)* (Outubro de 2011), 20–27.

[57] Feng, H., Shu, Y. Study on network traffic prediction techniques. *Proceedings. International Conference on Wireless Communications, Networking and Mobile Computing 2(3)* (2005), 995–998. doi: 10.1109/WCNM.2005.1544219.

[58] Kali Linux. Penetration Testing and Ethical Hacking Linux Distribution. Junho de 2018. Disponível em: https://www.kali.org/.