PONTIFÍCIA UNIVERSIDADE CATÓLICA DE CAMPINAS

CEATEC

FACULDADE DE ENGENHARIA ELÉTRICA

ARGEMIRO BEVILACQUA

DETECÇÃO E MITIGAÇÃO DE ANOMALIA NA CAMADA MAC EM REDES IEEE 802.11

PUC-CAMPINAS

2015

ARGEMIRO BEVILACQUA

DETECÇÃO E MITIGAÇÃO DE ANOMALIA NA CAMADA MAC EM REDES IEEE 802.11

Dissertação apresentada, como exigência para obtenção do Título de Mestre em Engenharia Elétrica, ao Programa de Pós-Graduação *Stricto Sensu* em Engenharia Elétrica da Pontifícia Universidade Católica de Campinas.

Orientador: Prof. Dr. Alexandre de Assis Mota.

PUC-CAMPINAS

2015

Ficha Catalográfica
Elaborada pelo Sistema de Bibliotecas e
Informação - SBI - PUC-Campinas

t621.2845 Bevilacqua, Argemiro.
B571d Detecção e mitigação de anomalia na camada MAC em redes IEEE 802.11 / Argemiro Bevilacqua. - Campinas: PUC-Campinas, 2015.
101p.

Orientador: Alexandre de Assis Mota.
Dissertação (mestrado) – Pontifícia Universidade Católica de Cam- pinas, Centro de Ciências Exatas, Ambientais e de Tecnologias, Pós-Graduação em Engenharia Elétrica.
Inclui bibliografia.

1. Sistemas de comunicação sem fio. 2. Telecomunicações. 3. Programação (Computadores). 4. Linux (Sistema operacional de computador). I. Mota, Alexandre de Assis. II. Pontifícia Universidade Católica de Campinas. Centro de Ciências Exatas, Ambientais e de Tecnologias Pós-Graduação em Engenharia Elétrica. III. Título.

22.ed. CDD – t621.3845

ARGEMIRO BEVILACQUA

Detecção e Mitigação de Anomalia na Camada MAC em redes IEEE 802.11

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro de Ciências Exatas, Ambientais e de Tecnologias da Pontifícia Universidade Católica de Campinas como requisito parcial para obtenção do título de Mestre em Gestão de Redes de Telecomunicações.
Área de Concentração: Gerência de Redes de Teleinformática. Orientador: Prof. Dr. Alexandre de Assis Mota.

Dissertação defendida e aprovada em 15 de dezembro de 2015 pela Comissão Examinadora constituída dos seguintes professores:

Prof. Dr. Alexandre de Assis Mota
Orientador da Dissertação e Presidente da Comissão Examinadora
Pontifícia Universidade Católica de Campinas

Profa. Dra. Lia Toledo Moreira Mota
Pontifícia Universidade Católica de Campinas

Prof. Dr. Paulo Batista Lopes
Universidade Presbiteriana Mackenzie

Dedico este trabalho à minha esposa Ana Maria

e aos meus filhos André e Lívia,

meus amores.

AGRADECIMENTOS

Agradeço pela compreensão e incentivo de meus entes queridos, em especial à minha esposa Ana Maria e meus filhos André e Lívia. Simplesmente maravilhosos.

Ao meu Orientador Professor Dr. Alexandre de Assis Mota pela orientação, esclarecimento, direcionamento e incentivo. Como líder do grupo de pesquisa institucional em Eficiência Energética, o Dr. Mota reuniu vários colegas de mestrado e professores nos laboratórios onde passamos tardes inteiras, muito produtivas.

À minha professora Drª. Lia Toledo Moreira Mota, pelos ensinamentos muito úteis ao desenvolvimento do meu tema.

A todos os professores do programa.

Agradeço especialmente ao amigo Claurem Paulus Ceolin Marques, Mestre da turma de 2013, que generosamente me acolheu com muita paciência e atenção no esclarecimento de todas as dúvidas com relação ao tema em que fui orientado a dar prosseguimento.

Aos colegas Lorenzo Coiado, Leandro Filiagi e todos pelo companheirismo e solidariedade.

A PUC Campinas pela bolsa de estudos e pelo execelente ambiente de aprendizado.

"Alguns de vocês, jovens foram me dizendo ...
- Hei Senhor, o que você quer dizer
"Que mundo maravilhoso?"
E todas aquelas guerras em todo o lugar?
Você chama isso de maravilhoso?
E quanto à fome e a poluição?
Isso não é tão maravilhoso também.
- Bem, que tal ouvir um velho Senhor por um minuto.
Me parece, não é o mundo que é tão ruim.
Mas o que estamos fazendo para ele.
E tudo que eu estou dizendo
é ver que mundo maravilhoso seria,
se apenas lhe dermos uma chance.
Amem queridos, amem.
Esse é o segredo, sim!
Se muito mais de nós amásse-mos uns aos outros.
Nós resolveríamos muitos problemas.
E então esse mundo seria maravilhoso.
É isso aí, o velho Senhor continua dizendo."

Louis Armstrong.

https://www.youtube.com/watch?v=D67lR7Qy_wk

RESUMO

O padrão IEEE 802.11 está presente em toda a parte como ponto de acesso à Internet sem fio. A maioria dos dispositivos utilizados é de baixo custo e não provê uma interface padronizada para gerenciamento remoto. Dispositivos móveis competindo pelo ponto de acesso podem causar uma degradação no desempenho da rede, conhecida como “Anomalia da MAC”. Neste estudo é apresentado um dispositivo com *firmware* modificado, o que possibilita a monitoração da qualidade de suas conexões sem fio em tempo real. Este *firmware* especial possui recursos de programação, permitindo que o ponto de acesso seja gerenciado remotamente. O Linux embarcado OpenWRT foi adotado como plataforma neste trabalho, em substituição ao *firmware* original. Foi escolhido um *hardware* de tamanho reduzido, de baixo custo e acessível. Através de programação, é possível coletar importantes parâmetros sobre a operação da rede sem fio, o que nos permite obter considerável controle sobre o ponto de acesso. Pontos de acesso à Internet sem fio são dispositivos roteadores e o Linux OpenWRT possui recursos de roteamento e controle de tráfego configuráveis e programáveis. Desta maneira, é possível adicionar o recurso de controle de tráfego ou políticas de roteamento através de programação. Neste trabalho foram coletados dados de vazão, tornando-se possível observar a anomalia da MAC, bem como identificar as estações ofensoras. Após isso, afim de mitigar o efeito danoso da anomalia ao desempenho da rede, foi utilizada a técnica *traffic shaping* para forçar um decréscimo de vazão nas estações ofensoras, restaurando assim a operação normal da rede. O objetivo deste trabalho é integrar os *scripts* de medição de vazão, identificação de ofensores e mitigação da anomalia, obtendo como resultado um dispositivo livre de quedas de desempenho devido à anomalia da MAC e que pode ser instalado em uma rede convencional para testes de campo.

Palavras chave: IEEE 802.11, Anomalia da MAC, Linux Embarcado, Gerenciamento de Rede.

ABSTRACT

IEEE 802.11 standard is largely used anywhere as a cheap way to access Internet, but the majority of devices used does not provide a standard way to manage them remotely. Mobile stations competing for the access point may cause a general wireless network failure, known as the "MAC Performance Anomaly". Here, it is presented an access point with modified firmware, that monitors wireless conections´s quality and has a programmable capability, allowing to do network management in a standardized manner. Embedded Linux "OpenWRT" was installed to replace the original firmware in a very cheap and reduced size hardware. By means of Bourne shell script programming, it is possible to collect all important operating parameters and data of the access point. From this, its possible to gain considerable control over it. Wireless access point devices are routers, while embedded Linux has routing capabilities. Thus, it is easy to implement traffic policies by means of Bourne shell script programming. Traffic shaping is one of the acces point´s capabilities successfuly tested and demonstrated in this work. MAC performance anomaly detection in IEEE 802.11 networks can be easily implemented by means of scripts as well. It was collected and plotted network throughput data, becoming possible to observe the MAC performance anomaly, identifying the offending stations. Furthermore, traffic shaping is applied in order to mitigate the effects of the anomaly, restoring in this manner, the normal network operation. The object of the current study is to integrate everything: measurements of operation data (network throughput in Mbit/s), detection of the MAC performance anomaly and perform its mitigation immediately. The ultimate goal is to install this device in a test field and therefore mitigate the MAC anomaly automatically.

Keywords: IEEE 802.11, MAC Anomaly, Embedded Linux, Network Management.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACK	=	*Acknowledge*
AP	=	*Access Point*
CCK	=	*Complementary Code Keying*
CPU	=	*Central Processing Unit*
CSMA/CA	=	*Carrier Sense Multiple Access with Collision Avoidance*
DCF	=	*Distributed Coordination Function*
DIFS	=	*DCF Interframe Space*
DSSS	=	*Direct Sequence Spread Spectrum*
EDCA	=	*Enhanced Distributed Channel Access*
FHSS	=	*Frequency-Hoping Spread Spectrum*
HCCA	=	*(Hybrid Coordination Function) Controlled Channel Access*
HCF	=	*Hybrid Coordination Function*
IEEE	=	*Institute of Electrical and Electronics Engineers*
IP	=	*Internet Protocol*
ISO	=	*International Standards Organization*
ISP	=	*Internet Service Provider*
LAN	=	*Local Area Network*
LED	=	*Light Emitting Diode*
MAC	=	*Media Access Control*
MAN	=	*Metropolitan Area Network*
MIMO	=	*Multiple Input Multiple Output*
NAV	=	*Network Allocation Vector*
OFDM	=	*Orthogonal Frequency-Division Multiplexing*
OSI	=	*Open Systems Interconnection*
PCF	=	*Point Coordination Function*
QoS	=	*Quality of Service*
RSSI	=	*Received Signal Strength Indicatior*

RTC	=	*Real Time Clock*
RTS/CTS	=	*Request to Send / Clear to Send*
SDM	=	*Spatial Division Multiplexing*
SDRAM	=	*Synchronous Dynamic Random-Access Memory*
SIFS	=	*Short Interframe Space*
SNR	=	*Signal-to-Noise Ratio*
STBC	=	*Space Time Block Coding*
TCP	=	*Transport Control Program*
TxBF	=	*Transmit Beam Forming*
UART	=	*Universal Asynchronous Receiver / Transmitter*
USB	=	*Universal Serial Bus*
Wi-Fi	=	*Wireless Fidelity*
WLAN	=	*Wireless Local Area Network*
WRT	=	*Wireless Router*

SUMÁRIO

1. INTRODUÇÃO

A importância da Internet e sua rápida expansão nos dias de hoje são fatos indiscutíveis. Até 1997, todos os projetos de rede local nas empresas eram feitos levando-se em conta somente a rede cabeada, padrão IEEE 802.3. A sigla IEEE significa: Instituto dos Engenheiros Elétricos e Eletrônicos, entidade de abrangência mundial, que dentre outras funções, padroniza os equipamentos de comunicação de dados. O padrão IEEE 802.3 contém as especificações para as camadas física e de enlace das redes cabeadas Ethernet. Devido à sua rápida expansão e a necessidade constante de mudanças de local das estações de trabalho conectadas, as redes locais cabeadas requerem um alto custo de manutenção para lançamento ou retirada de cabos.

A primeira rede local sem fio no padrão IEEE 802.11 foi lançada em 1997, tendo boa aceitação e expansão rápida (Kim, 2015). Vários equipamentos foram, então, conectados à rede local sem fio, denominada WLAN, (*Wireless Local Area Network*). Como exemplo, nas redes empresariais tem-se impressoras, computadores pessoais, câmeras filmadoras, leitores de código de barras, dentre outros dispositivos conectados neste tipo de rede. Já no ambiente residencial, pode-se encontrar, conectados às redes WLAN, dispositivos como *smart TVs*, *smart phones*, computadores pessoais, *tablets*, *video-games* e etc.

A rede local sem fio é denominada de rede de acesso à Internet, pois situa-se em suas bordas. O equipamento que dá suporte a esta rede é chamado de Ponto de Acesso ou *Access Point* (AP).

Nas redes sem fio, as estações cliente tem mobilidade e o ponto de acesso recebe um número variável de conexões. Ocorre uma disputa pelo acesso ao meio (ar) na camada de enlace, antes que as estações cliente estabeleçam uma conexão com o ponto de acesso. Pode-se esperar a conexão de qualquer tipo de equipamento, pois é uma rede cujos nós cliente são heterogêneos. Nesse cenário, algumas estações conectadas à rede sem fio podem apresentar uma conexão de baixa qualidade, o que resulta em uma taxa de vazão (Mbits/s) muito baixa. Isto pode

ocorrer, por exemplo, porque alguns dispositivos de *hardware* reduzido podem se conectar à rede, e podem não possuir uma interface para rede sem fio que implemente corretamente ou totalmente a especificação IEEE 802.11. Conexões de baixa qualidade podem ter origem também na localização desses dispositivos, como por exemplo: atrás de barreiras físicas que impeçam a propagação adequada do sinal de rádio ou muito distantes do limite de alcance do AP.

A disputa pelo acesso ao meio por uma estação com conexão de baixa qualidade e vazão muito menor do que a maioria das demais estações conectadas, pode gerar uma queda na vazão total da rede. Estações que estavam transmitindo a uma taxa alta, repentinamente têm sua taxa reduzida a níveis semelhantes ao da estação com conexão de baixa qualidade. Este fenômeno é bem difundido na literatura e denominado como "Anomalia da MAC" (Heusse *et al.,* 2003).

A Anomalia da MAC foi demonstrada pela primeira vez por Martin Heusse (Heusse *et al.,* 2003), utilizando modelagem e simulações, sendo que medições de dados reais comprovaram as simulações realizadas. A anomalia verificada na camada de enlace, sub-camada MAC de acesso ao meio, é explicada em detalhes, na seção 2.4, juntamente com o padrão IEEE 802.11 para redes sem fio.

Nesse contexto, o principal objetivo do presente trabalho é definir uma configuração de bancada de testes capaz de detectar e mitigar esta anomalia.

A Anomalia da MAC também é tratada em várias outras referências, como se segue. Na referência (Branquinho *et al.,* 2006), os autores emulam e mitigam a anomalia, usando medições de SNR (*Signal to Noise Ratio* – Relação Sinal-Ruído) em uma bancada com ambiente controlado em laboratório.

Na referência (Garroppo et al., 2006) foi possível isolar o tráfego direcionado a cada uma das estações, criando filas de pacotes em separado para cada uma e aplicando em seguida a técnica de *traffic shaping* sobre estas filas. Ou seja, foi possível dividir a banda da rede com igualdade entre as estações e, desta maneira, mitigar a anomalia da MAC. Para tanto, foi utilizado um AP com *firmware* especial.

A referência (Guirardello, 2008) demonstra a possibilidade do uso de QoS (*Quality of Service*) para mitigar a anomalia da MAC. Foi montada uma bancada experimental com duas estações cliente e um AP utilizando *firmware* modificado e antena de rádio especialmente controlada.

Já a referência (Peris, 2012) utiliza uma distribuição de Linux embarcado rodando em um ponto de acesso (DD-WRT Project, 2015). É aplicada a gerência de banda, um recurso encontrado no sistema operacional DD-WRT para mitigar a Anomalia da MAC, mas nenhum método para identificá-la é apresentado. Além do dispositivo de ponto de acesso, mais dois servidores conectados à rede cabeada são utilizados, sendo que todas as operações são realizadas manualmente.

Finalmente, a referência (Marques, 2013) apresenta um método matemático para detectar a anomalia da MAC, baseado em medições de vazão das estações conectadas ao ponto de acesso. Pela primeira vez, uma configuração comum e de baixo custo, facilmente encontrada no mercado é utilizada, o que possibilita a retirada da bancada do laboratório para execução de testes em campo, em um ambiente real de uso.

Neste contexto, o presente trabalho utiliza outros equipamentos e métodos para se obter um ponto de acesso auto gerenciável capaz de executar as funções de:

- medição de vazão da rede,
- detecção da anomalia e
- mitigação da anomalia

Estas funções devem ser executadas de maneira automática, integrada e livre de intervenção humana. Desta maneira, objetiva-se uma rede sem fio que opere de maneira autônoma, podendo ser utilizada em um ambiente do mundo real, minimizando os efeitos da anomalia.

A dissertação encontra-se organizada da seguinte maneira. O Capítulo 2 apresenta um histórico, descreve em detalhes o padrão IEEE 802.11 para redes de acesso à Internet sem fio e explica como ocorre a Anomalia da MAC. O Capítulo 3

aborda dois métodos para detecção da anomalia e identificação de ofensores: índice de sensibilidade (Marques, 2003) e o novo método desenvolvido e apresentado neste trabalho, o método do Coeficiente de Correlação Produto-Momento de Karl Pearson. No capítulo 4 é descrita a metodologia proposta, contendo a descrição dos equipamentos, *software* e método matemático utilizados. Os resultados dos testes e sua discussão são apresentados no capítulo 5. As sugestões para continuidade deste trabalho, limitações e conclusões finais são apresentadas no capítulo 6.

2. REDES SEM FIO PADRÃO IEEE 802.11

Redes locais de computadores cabeadas padrão Ethernet 802.3 eram, até pouco tempo, a única rede de acesso à Internet. Estas redes são conectadas a um ISP (*Internet Service Provider)*, onde se encontra um *gateway* que dá acesso à rede mundial. Com o crescimento acelerado do número de usuários da Internet e o surgimento de dispositivos portáteis como *lap-tops, tablets* e, principalmente, *smartphones,* as redes de acesso à Internet sem fio apareceram para dar suporte a estes novos dispositivos. Devido à sua ampla aceitação, a evolução das redes sem fio foi rápida, tendo várias *releases* em curto período de tempo (Kim, 2015).

2.1 Histórico

O padrão IEEE 802.11 para redes locais sem fio (WLAN) é um dos padrões de maior sucesso para sistemas de comunicação sem fio (Kim, 2015). Em 1980, O IEEE criou um comitê denominado LAN/MAN Standards Committee (LMSC), ou simplesmente 802. O comitê 802 cuida das especificações de uma família de padrões relativos a LAN/MAN, *(Local Area Network / Metropolitan Area Network),* englobando vários grupos de trabalho restritos às camadas de enlace e física do modelo OSI *(Open Systems Interconnect)* (ISO, 1989). A fim de definir um padrão mundial para interconexão de sistemas em redes de computadores, a ISO *(International Standards Organization)* definiu o padrão OSI. Para WLANs, a primeira especificação do IEEE foi publicada em 1997 (Kim, 2015), (IEEE Standards Association, 2015), (IEEE 802 Comitee Website, 2015) (IEEE 802 Standards, 2015). Inicialmente a especificação 802.11 definia apenas a taxa de 1 ou 2 Mbits/s operando na frequência de 2.4 GHz e usando a técnica de *Frequency Hopping Spread Spectrum* (FHSS), *Infra-Red* (IR) ou *Direct Sequence Spread Spectrum* (DSSS).

Mais tarde, decidiu-se expandir o padrão em dois grupos de trabalho: 2.4 GHz e 5 GHz, denominados de 802.11b e 802.11a, respectivamente. A codificação CCK - *Complementary Code Keying* foi introduzida no padrão 802.11b para suportar até 11 Mbits/s, enquanto que a codificação OFDM (*Ortogonal Frequency Division*

Multiplexing) foi aplicada ao padrão 802.11a, que suporta até 54 Mbits/s. Ambos adendos foram publicados em 1999 (Kim, 2015).

A idéia era ter dois tipos de padrões, um para cada tipo de aplicação, dependendo dos requisitos de maior ou menor taxa de vazão dos dados. Porém, devido ao alto custo, o padrão 802.11a foi pouco utilizado, ou somente usado pelas empresas, enquanto que o padrão 802.11b mais barato, teve uma demanda mais intensa, podendo-se identificar a necessidade de maior taxa de vazão para este padrão (Kim, 2015).

Em 2002, a fim de suportar a taxa de transferência de dados de 54 Mbits/s na banda de 2.4 GHz, o grupo de trabalho 802.11g decidiu adotar a mesma camada física e a mesma MAC *(Media Access Control)* da especificação do padrão 802.11a.

Com o aumento do uso da Internet para conteúdo multimídia, a demanda por taxas mais altas nunca parou de crescer. Como resultado, o comitê 802 aprovou a criação do grupo de trabalho 802.11n para desenvolver um novo adendo ao padrão 802.11, com vazão ainda maior. O objetivo do 802.11n era alcançar a vazão de 100 Mbits/s na camada MAC. Para tanto, o 802.11n empregou a técnica MIMO *(Multiple-Input Multiple Output*, incluindo SDM *(Spatial Division Multiplexing*) com até 4 fluxos ou *streams* de dados. A codificação STBC *(Space Time Block Coding)* foi adotada em conjunto com TxBF *(Transmit Beamforming)* (Kim, 2015).

Utilizando múltiplas antenas tanto no transmissor quanto no receptor, o SDM permite a multiplexação de múltiplos fluxos de dados através de dimensões espaciais, aumentando a vazão proporcionalmente à quantidade de fluxos de dados. Além disso, a tecnologia TxBF melhora a força do sinal recebido, enfatizando os modos dominantes de transmissão por canal (Kim, 2015).

Após ter completado o 802.11n em 2009, surge o padrão IEEE 802.11ac com aprimoramentos do método MIMO, liberando uma faixa de frequência de até 160MHz e até 8 fluxos de dados. Estas técnicas permitiram alcançar uma vazão na rede maior do que 1 Gbit/s. A tabela 1 mostra a evolução do padrão IEEE 802.11.

Tabela 1: *Releases* do padrão IEEE 802.11

Protocolo IEEE 802.11	Publicação	Frequência (GHz)	Banda (MHz)	Vazão (Mbits/s)	Canais MIMO	Modulação	Alcance (m)
1997	junho de 1997	2.4	22	1 ou 2	-	DSSS, FHSS	100
a	setembro de 1999	5	20	até 54	-	OFDM	120
b	Setembro de 1999	2.4	22	Até 11	-	DSSS	140
g	Junho de 2003	2.4	20	até 54	-	OFDM, DSSS	140
n	Outubro de 2009	2.4, 5	20, 40	Até 150	4	OFDM	140
ac	Dezembro de 2013	5	20, 40, 80, 160	Até 780	8	OFDM	115

Outro marco importante ocorreu em 1999, com a criação da Wi-Fi Alliance, que é uma associação comercial de empresas que criou a marca e o logotipo *Wi-Fi (Wireless Fidelity)* para prover testes e certificação de equipamentos no padrão de redes sem fio IEEE 802.11. Assim, os termos Wi-Fi, padrão IEEE 802.11 e WLAN estão fortemente ligados.

2.2 O Padrão IEEE 802.11

O padrão IEEE 802.11 contém uma série de especificações para a camada de enlace e para a camada física do modelo OSI (IEEE Standards Association, 2015). A sub-camada MAC mostrada na figura 1, que controla o acesso ao meio é o objeto de estudo deste trabalho.

Nas comunicações sem fio, o meio de transmissão é o ar. Como tem-se apenas um canal de rádio disponível no ponto de acesso, as estações cliente e o próprio

ponto de acesso tem que esperar a sua vez para transmitir. Sendo assim, a comunicação é do tipo *half-duplex,* (não ocorrem transmissões e recepções simultâneamente) e ocorre uma disputa pelo meio de transmissão toda vez que um pacote precisa ser transmitido. O método de acesso adotado pelo padrão IEEE 802.11 é o CSMA/CA (*Carrier Sense Multiple Access with Collision Avoidance).*

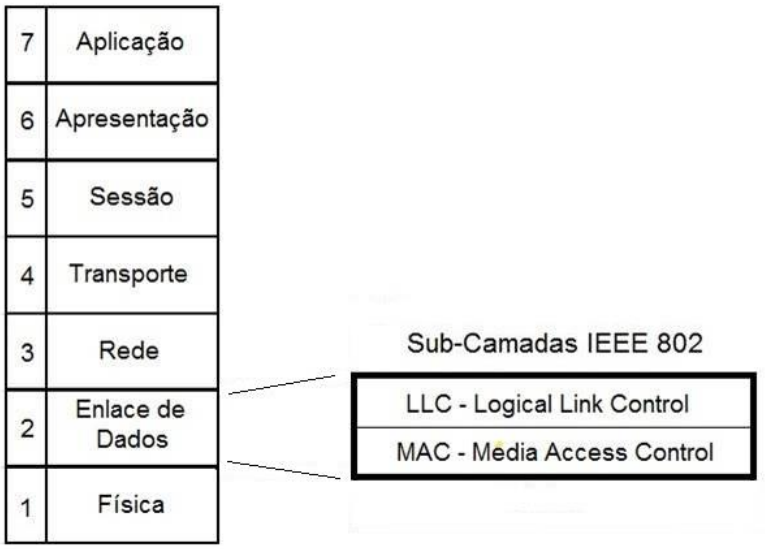


Figura 1: Modelo OSI com sub-camada MAC em destaque.
Extraída de (Marques, 2013)

O protocolo primário que implementa o método de acesso ao meio é o DCF *Distributed Coordination Function* (Bianchi, G., 2000). O DCF é o protocolo básico mais utilizado e aplica-se a todas as versões IEEE 802.11: a, b, g, n e ac. Existem outros protocolos tais como o PCF (*Point Coordination Function)* usado somente no ponto de acesso quando este é configurado no modo *infrastructure*. Outros protocolos opcionais como EDCA (*Enhanced Distributed Channel Access*) e HCF (*Hybrid Coordination Function*) podem ser implementados em substituição ao DCF, mas não são muito utilizados. A figura 2, ilustra a pilha de protocolos de acesso ao meio para a especificação IEEE 802.11.

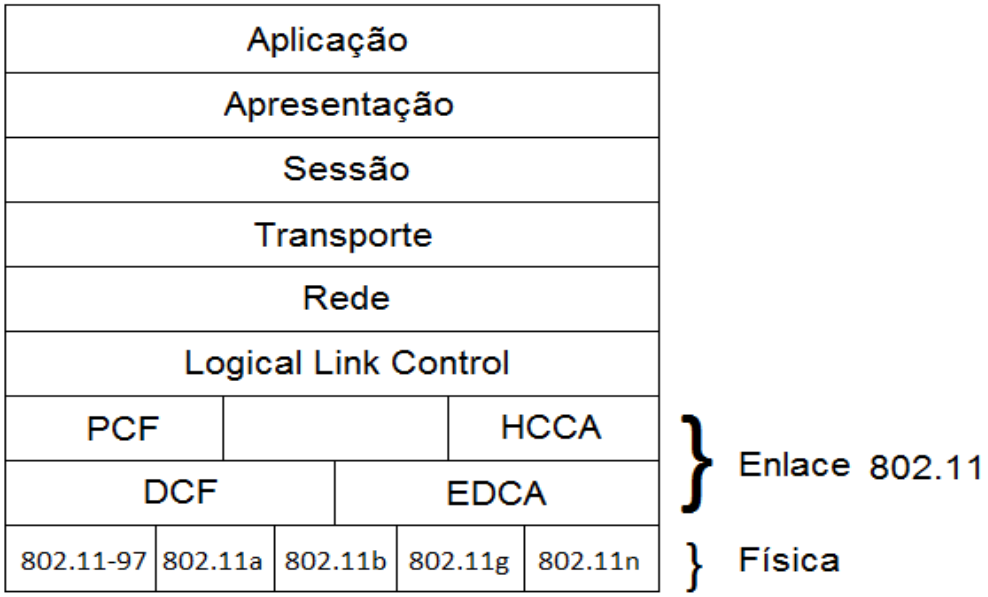


Figura 2: Protocolos de Acesso ao Meio (IEEE Standards Assoc, 2015)

O mecanismo de RTS/CTS *(Request to Send / Clear to Send)* é utilizado em conjunto com o protocolo DCF. Os eventos acorrem sobre uma linha do tempo imaginária conforme mostram as figuras 3 e 4.

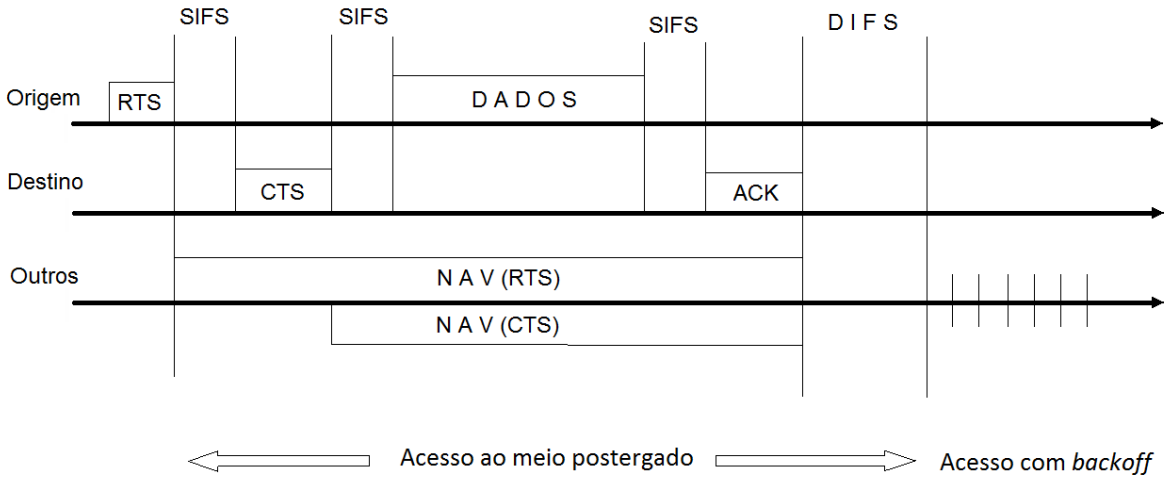


Figura 3: Mecanismo RTS / CTS (Garropo, 2006)

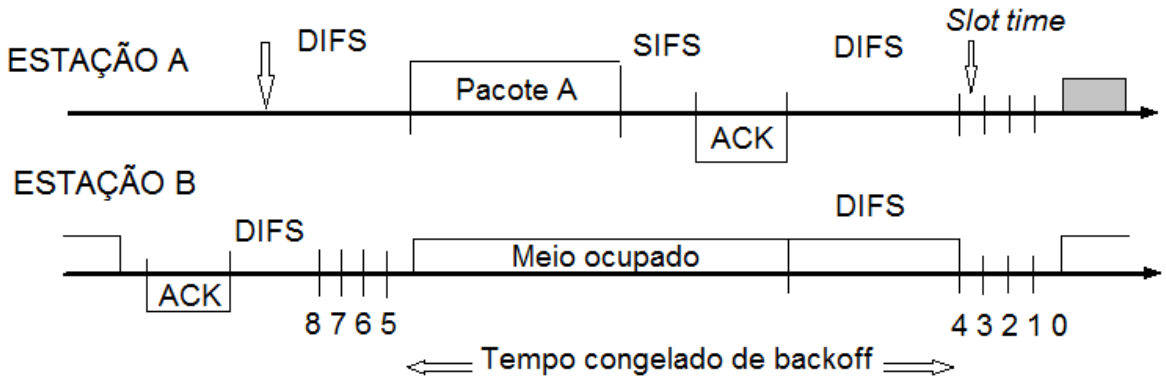


Figura 4: Protocolo DCF (Garropo, 2006)

No modo RTS/CTS, antes de transmitir, a estação envia um *frame* de controle do tipo RTS para reservar o canal. A estação de destino reconhece o recebimento do RTS e envia um CTS como resposta. Em seguida, tem início a transmissão normal que tem como resposta o envio de um *frame* de controle do tipo ACK (acknowledge).

Como uma eventual colisão poderá ocorrer somente durante a transmissão do frame especial RTS e é detectada através da falta de uma resposta do tipo CTS, o mecanismo CTS/RTS possibilita o incremento do desempenho do sistema através da redução do tempo gasto em colisões quando longas mensagens são transmitidas.

Na figura 4, que apresenta o protocolo DCF, a estação com um novo pacote para transmitir monitora a atividade do canal. Se o canal ficar livre por um período de tempo igual a DIFS *(Distributed Interframe Space),* então a estação transmite. Caso contrário, se for verificado que o canal está ocupado, a estação continua monitorando o canal, até que fique livre por um período igual a DIFS. Neste ponto, a estação entra em um intervalo de espera ramdômico chamado de *backof,* antes de transmitir, para minimizar colisões com os pacotes sendo transmitidos pelas outras estações da rede.

Além disso, a fim de evitar a retenção do canal, a estação tem que esperar por um tempo randômico de *backoff* entre a transmissão de dois pacotes consecutivos, mesmo que o meio fique livre por um intervalo de tempo igual a DIFS.

Objetivando melhorar sua eficiência, o método DCF emprega tempos de *backoff*, após cada intervalo de tempo DIFS, sendo que as estações só podem transmitir no início de cada intervalo de *backoff*. Estes intervalos de tempo são iguais ao tempo gasto por qualquer estação para detectar a transmissão de um pacote por uma outra estação e dependem da camada física.

Um *frame* de controle to tipo ACK é transmitido pela estação de destino para sinalizar a recepção do pacote com sucesso, imediatamente ao final de seu recebimento, pois a estação transmissora não é capaz de detectar colisões com seu próprio pacote, no tempo em que está transmitindo. O intervalo de tempo entre o fim da mensagem e a respectiva resposta ACK, é chamado de SIFS (*Short Interframe Interval*). Como o intervalo de tempo correspondente à soma do SIFS mais o tempo de propagação do ACK é sempre menor que o DIFS, nenhuma outra estação será capaz de detectar a condição de canal livre, até que a transmissão do ACK termine. Se não houver um ACK, significa que o quadro não foi bem recebido pela estação de destino. O tempo de DIFS se esgota, então uma nova disputa pelo acesso ao meio se inicia. A estação que ainda não recebeu o ACK decide pedir uma retransmissão, depois de conseguir acesso ao meio.

2.3 Estrutura de Redes sem Fio

Uma rede sem fio pode ser configurada em dois modos básicos: *Infrastructure Mode* ou *Mesh Mode*. No modo *infrastructure*, uma estação central controla toda a rede, normalmente do ponto de acesso à Internet. No modo *Mesh*, uma estação pode se conectar a uma outra qualquer, em modo ponto-a-ponto. Se uma terceira estação entrar na rede, tem-se uma topologia multiponto sem coordenação central.

A estação central, denominada de ponto de acesso à Internet, é um roteador que faz a conexão entre a rede sem fio e uma rede cabeada. Através da rede cabeada é feito o acesso à Internet por meio de um ISP *(Internet Service Provider)*. A figura 5 ilustra a estrutura utilizada em redes sem fio.

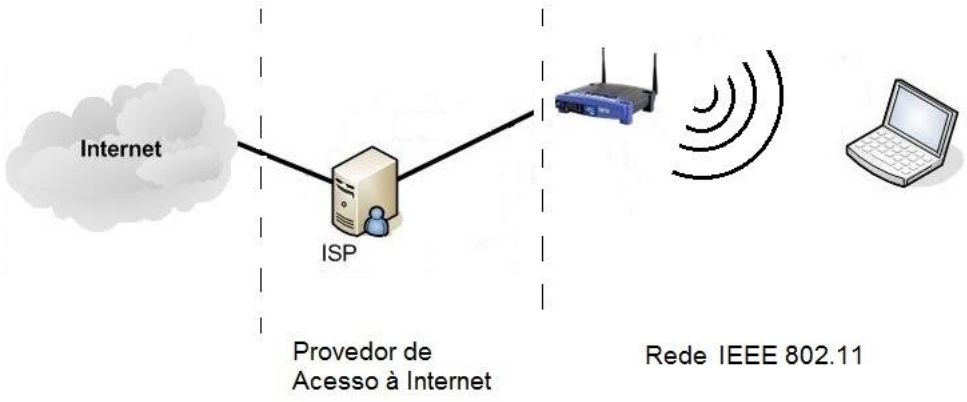


Figura 5: Estrutura de Redes sem Fio

O meio de transmissão nessas redes é o ar com apenas um canal. Sendo assim, o modo será sempre *half-duplex.* As estações, incluindo aqui o ponto de acesso, precisarão disputar o meio para poder transmitir. Dependendo da distância entre as estações se comunicando, as medidas de SNR *(Signal to Noise Rate)* e RSSI *(Received Signal Strength Indicator)* da rede podem variar. Com base nisto, e caso ocorram perdas de pacotes, a taxa de vazão é ajustada automaticamente pelo AP.

2.3.1 Roteamento em Redes IEEE 802.11

O ponto de acesso (AP) é o núcleo da rede sem fio WLAN. Além de prover comunicação sem fio para as estações cliente conectadas, cumpre a importante função de roteamento e controle de fluxo dos dados. Em uma rede TCP/IP *(Transport Control Protocol / Internet Protocol)*, computadores denominados roteadores ou *gateways* provêem todas as interconexões entre redes físicas (Comer, 1995). Frequentemente, as redes sendo interconectadas, não operam na mesma taxa de transmissão e/ou recepção.

Sendo assim, faz-se necessário um controle de fluxo dos dados, a fim de compatibilizar as diferentes taxas de vazão dos dados. A camada de transporte do protocolo TCP/IP é responsável pelo controle de fluxo que é processado pelo roteador da seguinte maneira: os pacotes recebidos são armazenados em *buffers* de entrada, copiados para *buffers* de saída, onde aguardarão por uma oportunidade para serem

transmitidos, de acordo com a política em vigor no roteador, ou com a disponibilidade do meio de transmissão.

Caso haja sobrecarga dos *buffers*, os pacotes entrantes podem ser perdidos. Então, é enviada uma mensagem de controle de fluxo, para que o transmissor diminua a sua taxa de transmissão e ao mesmo tempo uma solicitação para que sejam retransmitidos os pacotes perdidos. Uma descrição completa sobre controle de fluxo pode ser encontrada em (Comer, 1995), (Kurose, 2006).

Na configuração de rede apresentada aqui, observa-se 3 fluxos de dados: local entre as estações sem fio, *upstream* que é o fluxo em direção à Internet e *downstream*, que é o fluxo originado na Internet com destino à rede local.

Os roteadores, tendo armazenados os pacotes em *buffers*, são capazes de aplicar um processamento adicional sobre estes, antes de retransmiti-los. Através dos campos de cabeçalho TCP/IP, estes pacotes podem ser identificados e classificados em filas especiais. Assim, é possível aplicar políticas de controle de fluxo sobre determinados tipos de tráfego, ou roteá-los para destinos pré-definidos. Através de configuração e programação, o Linux embarcado permite-nos implementar tais políticas (Hubert, 2015).

2.4 A Anomalia da MAC em Redes IEEE 802.11

Conforme demonstrado na referência (Heusse, 2003), o protocolo DCF que governa a MAC, concede igual oportunidade de acesso ao meio à todas as estações cliente, mesmo que alguma delas tenha uma qualidade de conexão bem inferior às demais. A qualidade de uma conexão é definida por parâmetros de QoS *(Quality of Service)* tais como intensidade do sinal de rádio *(RSSI - Radio Signal Strength Indicator),* relação sinal-ruído *(SNR – Signal to Noise Ratio)* e atrazo/perdas dos pacotes.

Sendo forçada a retransmitir seus quadros muito frequentemente devido às perdas de pacotes, operando a uma vazão bem inferior do que as demais, a estação com sinal ruim levará muito mais tempo para transmitir seus dados.

Esta demora força as demais estações a uma longa espera, incluindo o próprio ponto de acesso. Quando o meio estiver livre para uma estação normal transmitir, esta rapidamente transmite apenas um quadro e começa a uma longa espera por uma nova oportunidade.

Devido à demora para a liberação do meio, causada pela estação em condições ruins de propagação (denominada estação ofensora), as estações com boas condições de propagação tem suas vazões diminuídas, incluindo o ponto de acesso. Como consequência, pode-se verificar um decréscimo na vazão total da rede, passando a ser similar à vazão da estação ofensora. Na figura 6 é apresentado um cenário com anomalia instalada.

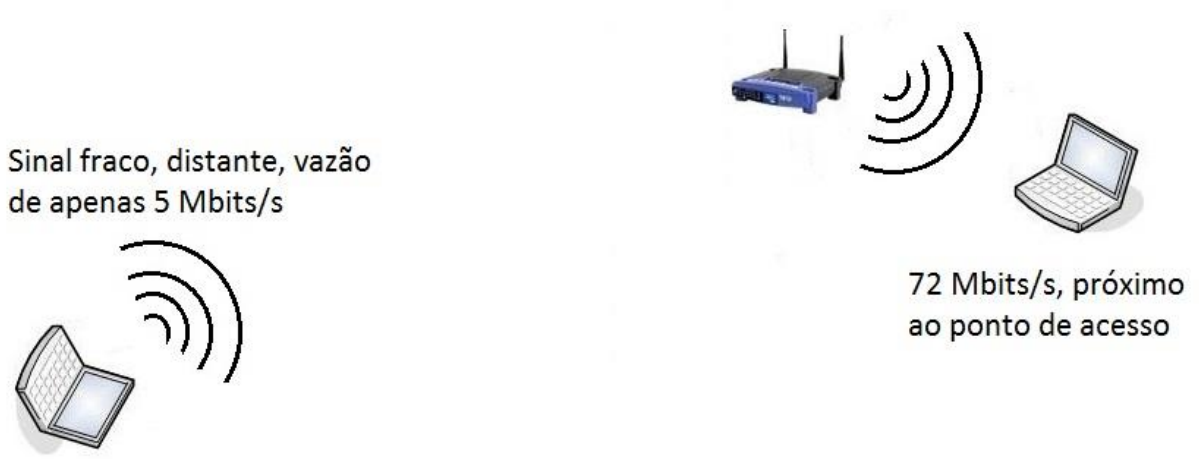


Figura 6: Cenário com Anomalia.

O padrão IEEE 802.11n, uma geração mais recente de redes sem fio, adicionou várias tecnologias à camada física, a fim de melhorar a cobertura do sinal, aumentando significativamente as taxas de vazão. Contudo, nenhuma mudança ocorreu no núcleo da camada de enlace: a sub-camada MAC. Assim, a anomalia ainda pode ser observada para este padrão (Zhang *et al.,* 2008), (Shrivastava *et al.,* 2008), (Lopez-Aguilera *et al.,* 2010), (Cheng Yan-hong *et al.,* 2007).

2.4.1 Mitigação da Anomalia

Devido ao tipo de tráfego que passa atualmente pela rede, gerado em sua maioria por aplicações sensíveis ao atraso ou perda de pacotes, uma eventual anomalia na transmissão ou recepção dos dados, poderia facilmente tornar a rede inviável para uso. A figura 6 apresenta uma recente estatística tomada com relação ao tipo de tráfego observado na Internet. Aplicações tais como voz sobre IP, música ou *streamming* de vídeo são sensíveis à qualidade da conexão. Uma vez que os pacotes são armazenados em *buffers* de entrada e saída nas interfaces de rede, para depois serem apresentados à aplicação, um atraso pequeno e constante pode ser tolerado, sem prejuízo da comunicação. Por outro lado, se este atraso for variável (*jitter*) poderá inviabilizar aplicações tais como voz ou *streamming* de vídeo (Lorenz, 2009).

	Upstream		Downstream		Aggregate	
Rank	Application	Share	Application	Share	Application	Share
1	BitTorrent	47.55%	Netflix	32.69%	Netflix	29.03%
2	HTTP	11.45%	HTTP	17.48%	HTTP	16.59%
3	Netflix	7.69%	YouTube	11.32%	BitTorrent	13.47%
4	Skype	4.27%	BitTorrent	7.62%	YouTube	9.90%
5	SSL	3.57%	Flash Video	3.41%	Flash Video	3.04%
6	Facebook	2.19%	RTMP	3.12%	RTMP	2.81%
7	PPStream	1.73%	iTunes	3.05%	iTunes	2.69%
8	YouTube	1.64%	Facebook	1.78%	SSL	1.96%
9	Xbox Live	1.31%	MPEG	1.72%	Facebook	1.84%
10	Teredo	1.25%	SSL	1.69%	MPEG	1.49%
	Top 10	**82.63%**	**Top 10**	**83.88%**	**Top 10**	**82.83%**

SOURCE: SANDVINE NETWORK DEMOGRAPHICS

sandvine

Figura 7: Principais tipos de tráfego na Internet (Sandvine, 2011)

Diante do exposto, faz-se necessário minimizar os efeitos da anomalia imediatamente após o momento em que esta ocorrer. No capítulo 1, são citadas técnicas para mitigação da anomalia, que atuam nas camadas de enlace e física.

No presente trabalho, foi utilizada uma técnica de *traffic shapping*, que atua no roteador, camada de rede. Aplica-se uma política sobre o tráfego ofensivo, o que força

uma diminuição no seu fluxo de dados. Uma vez detectada a grande perda de pacotes que são descartados por esta política, o mecanismo de controle de fluxo da camada de transporte entra em ação, fazendo com que a estação ofensora baixe a sua vazão e ao mesmo tempo retransmita os pacotes descartados pelo roteador (Kurose, 2006). Como nenhum pacote será perdido, a estação ofensora ainda terá oportunidade de continuar transmitindo/recebendo. Não é aplicado *traffic shaping* sobre o tráfego das estações não ofensoras.

Contudo, antes de mitigar a anomalia, é necessário detectar a sua ocorrência e identificar as estações ofensoras, conforme será visto no capítulo 3.

3. O PROBLEMA DA IDENTIFICAÇÃO DO OFENSOR

A rede cabeada padrão IEEE 802.3 (Ethernet), possui conexões confiáveis, apresentando alta qualidade, na maioria das vezes. O mesmo não se pode afirmar sobre as rede sem fio WLAN que além de depender do sinal de rádio, tem suas estações em constante movimento, o que pode degradar a transmissão do sinal.

De acordo com (Rappaport, 2002), a propagação do sinal de rádio pelo ar depende diversos fatores tais como: distância, obstáculos entre transmissor e receptor, interferências de outras fontes de rádio, alterações no meio como por exemplo umidade e etc. O sinal de rádio empregado nas redes IEEE 802.11 nem sempre é constante e de boa qualidade. Portanto, um sinal de rádio atenuado pode causar uma comunicação de dados de baixa qualidade entre a estação e o ponto de acesso da rede sem fio, podendo ocasionar o fenômeno da Anomalia da MAC nessas redes.

Segundo a referência (Marques, 2013), para que se possa mitigar a anomalia, faz-se necessário primeiro identificar as estações ofensoras. Vários trabalhos apontam soluções para mitigação da anomalia, mas nenhum apresenta uma maneira para identificação dos ofensores. Nessas referências (Branquinho, 2006) (Guirardello, 2008) e (Peris, 2012), para demonstrarem a mitigação da anomalia, esta foi simulada, conhecendo-se previamente os ofensores. A referência (Marques, 2013) foi a primeira a propor um método para identificação de ofensores em redes IEEE 802.11. Pode-se definir como estação potencialmente ofensora, uma estação que possua uma ou mais das seguintes características (Marques, 2013):

- Potência do sinal recebido ou RSSI (*Received Signal Strength Indicator)* muito baixa.
- Vazão muito abaixo das demais estações da rede.
- Obsolescência ou pouca robustez, isto é, *hardware* reduzido que não implemente a pilha OSI por completo.
- SNR *(Signal-to-Noise Rate)* desfavorável à transmissão dos dados.

- Distância entre a estação e o ponto de acesso no limite de alcance do sinal, de acôrdo com especificações dos respectivos fabricantes.

Mas estes parâmetros não são suficientes para que se possa dizer com certeza, qual é a estação ofensora. A seguir (itens 3.1 e 3.2) são apresentados dois métodos matemáticos para identificação de ofensores com base em medições de vazão das estações conectadas. O primeiro encontra-se descrito na referência (Marques, 2013), e o segundo, é um método proposto e implementado computacionalmente neste trabalho.

3.1 Índice de Sensibilidade

O Índice de Sensibilidade proposto em (Marques, 2013), consiste em se analisar a sensibilidade da rede em relação a variações de níveis de vazão de uma estação em particular. Se uma pequena variação na taxa de vazão de uma estação causar uma grande alteração na taxa de vazão total da rede em sentido oposto, então esta é uma estação ofensora.

O cálculo do Índice de Sensibilidade de uma estação em relação à vazão total da rede é dado pela Equação (1):

$$S_i = \frac{\Delta V_i}{\Delta V_T} \qquad (1)$$

Onde ΔV_i é a variação na vazão da estação i e ΔV_T é a variação na vazão total da rede, que ocorrem entre os instantes inicial e final do intervalo de amostragem.

Consideremos $\Delta V_T = \Delta V_i = 10s$. Se após decorridos 10s ocorrer uma pequena queda na vazão da estação em análise e no mesmo instante for observado um aumento significativo na vazão total da rede, então a estação em análise deve ser considerada como ofensora. A recíproca é verdadeira. Caso neste intervalo de 10s ocorrer um pequeno aumento na vazão da estação em análise, acarretando uma queda desproporcional na vazão total da rede, será verificada também a ofensividade.

Os índices de sensibilidade da rede calculados em relação a estações ofensoras apresentam valores baixos e na maioria das vezes negativos.

O índice de sensibilidade é uma medida instantânea tendo seu cálculo feito em curtos intervalos de tempo. No exemplo dado, este intervalo é de 10s. Para intervalos de tempo mais extensos, considerando-se por exemplo o tempo total da massa de dados, o índice de sensibilidade não é apropriado, pois as pequenas oscilações das vazões seriam perdidas.

3.2 O Coeficiente de Correlação de Pearson

Este trabalho propõe a utilização do Coeficiente de Correlação Linear Produto-Momento, definido por Karl Pearson e, por essa razão conhecido na literatura como Coeficiente de Correlação de Pearson (Larson, 2010), para a identificação das estações ofensoras em uma rede IEEE 802.11.

A similaridade existente entre o comportamento de duas variáveis pode ser quantificada através do Coeficiente de Pearson, que é adimensional. Seus valores se situam entre -1, zero e +1.0, refletem a similaridade entre dois conjuntos de dados.

O Coeficiente de Correlação de Pearson é representado pela letra “ρ” (Larson, 2010). Sua interpretação é subjetiva para valores não exatos, ou seja, diferentes de -1, zero ou +1. Contudo, há que se adotar uma escala para os valores não exatos, pois na prática dificilmente encontra-se valores exatos como resultado. Não há um consenso na literatura, neste trabalho foi adotada a seguinte escala (Britto, 2009):

- $\rho = +1$ significa duas curvas com comportamento muito similar.
- $\rho = -1$ significa duas curvas com comportamento muito similar em sentido inverso.
- $0.7 < \rho < 1$ significa uma correlação acentuada entre duas curvas, positiva.
- $0.3 < \rho < 0.7$ significa uma correlação moderada entre duas curvas, positiva.
- $0 < \rho < 0.3$ significa uma correlação discreta entre duas curvas, positiva.
- $\rho = 0$ ou próximo de zero, não há correlação.
- $\rho < 0$ convenciona-se as mesmas intensidades acima, para correlações negativas.

O coeficiente “ρ” corresponde a um índice de similaridade entre duas curvas. Ou seja, se ρ = 1, tem-se duas curvas com comportamento bastante similiar, se ρ é próximo de 1, tem-se curvas semelhantes. Por outro lado, se “ρ” for negativo, tem-se curvas com comportamento bastante similar, porém em sentido oposto. Isto significa que quando uma aumenta a outra diminui na mesma proporção.

O cálculo desse coeficiente é dado pela Equação (2):

$$\rho = \frac{Cx,y}{Sx\ Sy}, \quad \rho \in [-1, 1] \qquad (2)$$

Onde Cx, y é a Covariância entre as variáveis x e y; Sx é o desvio padrão da variável x e Sy é Desvio padrão da variável y.

Nesse método, proposto nesta dissertação, os dados de entrada utilizados para identificação da estação ofensora são representados na forma de dois vetores: um, contendo a vazão (Mbits/s) de uma das estações e, outro, a vazão total da rede. Assim, é calculado o Coeficiente de Correlação de Pearson para todas as estações, uma a uma, em relação à vazão total da rede.

A partir da observação dos resultados descritos na referência (Marques, 2013), que apontam que uma pequena variação na taxa de vazão de uma estação ofensora causa uma grande alteração na vazão total da rede (sendo que essas variações são em sentidos opostos), espera-se que, se o Coeficiente de Correlação de Pearson (CCP) for negativo, então a estação em questão é uma ofensora.

Nesse método, o valor do CCP pode ser considerado ainda como um indicativo da “força” da ofensividade. Se o CCP for positivo, considera-se que a estação não é ofensora.

4. METODOLOGIA PROPOSTA

A metodologia deste trabalho compreende as seguintes etapas:

- Identificação automática das estações ofensoras, utilizando o Coeficiente de Correlação de Pearson, conforme descrito anteriormente.
- Mitigação da Anomalia da MAC de forma automática, utilizando *traffic shaping* para forçar uma queda de vazão na estação ofensora (seção 4.1) e, consequentemente, restaurando a operação normal da rede (Garropo, 2006).

Para tanto, foi implementada uma bancada de testes para a verificação dos métodos de identificação de ofensores e mitigação da Anomalia propostos neste trabalho.

O modelo de bancada apresentado em (Marques, 2013) foi escolhido como ponto de partida. Vários aperfeiçoamentos foram adicionados para a construção do novo ponto de acesso. O resultado foi um ponto de acesso barato, acessível, gerenciável e que pode ser utilizado para testes de campo em um ambiente do mundo real. A tabela 2 mostra as principais diferenças entre ambos os sistemas a bancada utilizada neste trabalho e a bancada utilizada em (Marques, 2013).

Tabela 2: Comparação entre as duas bancadas

Bancada deste trabalho	**Bancada de (Marques, 2013)**
Apenas uma peça de tamanho reduzido	CPU tipo torre, monitor, teclado e *mouse*
Custa R$ 70,00, fácil de encontrar	Custa aproximadamente R$ 1.000,00
Linux embarcado, funcionalidade específica	Linux completo com milhares de aplicações.
Wi-Fi é um recurso nativo	*Wi-Fi* tem que ser adaptado
Suporte ao protocolo SNMP	Suporte ao protocolo SNMP
Eficiência energética (5,4W)	Alto consumo de energia (300W)
4Mb de memória *Flash* e 32Mb de RAM	4Gb de RAM e 250Gb de disco fixo
Clock de CPU 400MHz	*Clock* de CPU 2.4GHz
Software livre, código fonte aberto	*Software* livre, código fonte aberto
Recursos de roteamento	Recursos de roteamento

Muito simples	Estação de trabalho Linux completa
Ideal para testes de campo	Difícil transporte e instalação
Documentação acessível	Documentação acessível
Permite controle de tráfego	Permite controle de tráfego
Padrão IEEE 802.11n com MIMO	Padrão IEEE 802.11b
Taxa de transferência de até 300Mbps	Taxa de transferência de até 11Mbps
Alcance de até 100m	Alcance de até 70m
Linguagens C, Lua e Bourne shell	Linguagens C, Lua, Bourne shell e outras

As principais contribuições deste trabalho vão muito além de melhorias básicas na bancada tais como novo *hardware* e sistema operacional conforme descrito na seção 4.1. Assim como em (Marques, 2013), também foi utilizada a plataforma Linux, apresentada em detalhes na seção 4.1.1. Programas similares aos de (Marques, 2013), também foram desenvolvidos a fim de coletar dados sobre as conexões das estações clientes. Dados tais como vazão em Mbits/s e RSSI *Radio Signal Strenght Indicator* em dBm podem ser coletados das estações clientes. A coleta de dados é descrita em detalhes nas seções 4.2, 4.3 e 4.4.

Após coletar os dados de vazão das estações conectadas, é possivel analisar a qualidade das conexões, para saber se uma dada estação é ofensora ou não. Uma nova abordagem foi desenvolvida para o processamento dos dados utilizando-se o CCP para identificar as estações ofensoras. Este método foi codificado em linguagem "Lua" (Ierusalimschy, 2015) e é explicado em detalhes na seção 3.2.

Um *script* codificado em linguagem Bourne shell foi testado com sucesso para efetuar a mitigação da anomalia na sub-camada MAC. Foram utilizados os recursos de controle de tráfego do Linux embarcado OpenWRT, rodando no ponto de acesso (OpenWRT *Project*, 2015). A técnica de *traffic shaping* foi utilizada para reduzir as vazões das estações ofensoras. Após aplicar *traffic shaping*, observou-se a imediata recuperação do nível original de vazão total da rede. Esta técnica e os *scripts* são explicados em detalhes na seção 4.4 (Hubert, 2015), (Garroppo, 2006).

Foram tomadas medidas de até 7 estações cliente conectadas a um ponto de acesso, sendo que 2 foram identificadas como ofensoras.

4.1 Estrutura da Nova Bancada

O novo ponto de acesso padrão 802.11n teve seu *firmware* substituído por um Linux embarcado (OpenWRT, 2015), o que possibilitou o uso de programação para a coleta dos dados de vazão e aplicação imediata do modelo em tempo real de operação da rede sem fio.

A bancada de testes é composta de um AP *(Access Point)* comum, de tamanho reduzido, portátil e que pode ser levado a campo para testes. O modelo escolhido foi o TL-WR841ND marca TP-LINK, que consta de uma lista de equipamentos compatíveis com o OpenWRT (OpenWRT, 2015). O *firmware* que originalmente rodava no AP foi totalmente substituído por um Linux embarcado, o OpenWRT versão “Barrier Breaker 14.07, LuCI Trunk 0.12+svn-r10530”, *kernel* versão 3.10.49. Uma descrição detalhada sobre o *hardware* e *software* do AP escolhido pode ser encontrada em (OpenWRT, 2015). Para efetuar conexões ao ponto de acesso, foram utilizados os equipamentos mais comumente utilizados para acesso à Internet sem fio: *lap-tops* e *smartphones*. Os *lap-tops* rodam Windows 7 e os *smartphones* rodam Android (Figura 7).

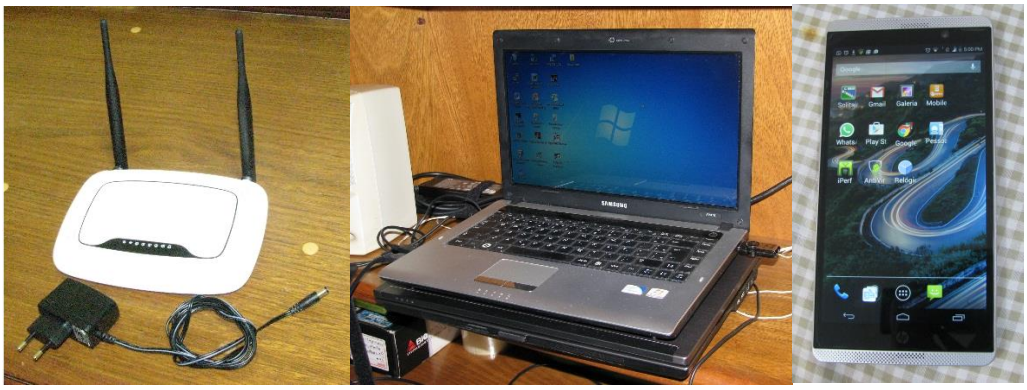


Figura 8: Equipamentos Utilizados nos Testes

O ponto de acesso é um roteador e o Linux embarcado OpenWRT possui todos os recursos de *software* necessários para medição e controle de tráfego (Hubert,

2015; OpenWRT, 2015). Além disso, o Linux provê recursos de programação e vários pacotes de *software* disponíveis para baixar e instalar gratuitamente. Foram utilizadas duas linguagens de programação em scripts: Bourne shell para medição de vazão e *traffic shaping* enquanto que a linguagem Lua (Ierusalimschy, 2015) foi utilizada para cálculo do CCP. Estas linguagens fazem parte do sistema operacional e já vêm embutidas quando instalamos o OpenWRT.

4.1.1 Linux Embarcado para Roteadores Wi-Fi

Conforme descrito em (Hallinan, 2006), sistemas Linux embarcados possuem vários atributos chave. Por exemplo, se o sistema sofre uma interrupção de energia, ele poderá se auto reiniciar, não dependendo de intervenção humana, agindo como um sistema autônomo. A interface homem-máquina, na maioria das vezes, resume-se a apenas alguns LED´s e uma interface serial.

Sistemas embarcados não precisam sempre ter tamanho reduzido. Eles podem se extender a vários "racks" em tamanho, como o são os sistemas para armazenamento de dados em massa. Sistemas Linux dedicados são amplamente utilizados para controlar este tipo de equipamento. Eles são chamados de "*appliances"*.

Sistemas Linux embarcados podem também ser encontrados em tamanho reduzido. Algumas características típicas de sistemas embarcados, (exceto *smartphones* e *tablets* que são sistemas de propósito geral) são:

- Limitação de recursos: pouca memória e nenhum disco rígido.
- Interface homem-máquina muito simples ou até mesmo nenhuma.
- Contém um microprocessador de propósito geral.
- Aplicações já vem embutidas, não são instaladas pelo usuário.
- São apresentados prontos para uso, com todo o *hardware* e *software* pré-instalados.
- Não preveem a ocorrência intervenção humana, normalmente são sistemas autônomos.

- Especialmente projetados para uma determinada finalidade.
- Não são de propósitos gerais, como a maioria dos computadores.
- Consumo baixo de energia ou alimentados por baterias.

O ponto de acesso do exemplo apresentado na figura 8, contém *on-board:*

- Um processador de 32 bits como seu componente central. Normalmente, o processador executa várias outras funções além daquelas previstas em uma CPU convencional. Em nosso exemplo temos integrados uma UART para porta serial, uma interface USB e uma controladora Ethernet.
- Programas e dados não voláteis são armazenados na memória *flash*. A memória principal usa tecnologia SDRAM *Synchronous Dynamic Ramdom-Access Memory,* podendo conter poucos ou centenas de Megabytes, dependendo da aplicação.
- Um módulo de relógio de tempo real RTC *Real Time Clock*, que mantém data, hora, minutos e segundos através de uma bateria.
- Este exemplo inclui três interfaces: uma USB, uma Ethernet e uma serial padrão RS-232 para acesso ao sistema operacional. O conjunto de chips 802.11 implementa a função de modem para comunicação sem fio.

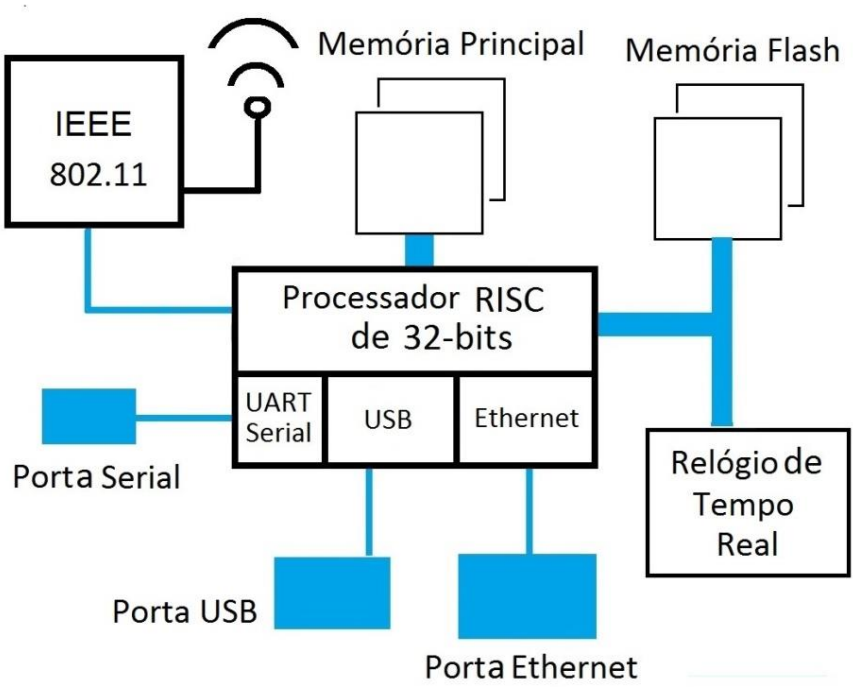


Figura 9: Estrutura de um Ponto de Acesso Sem Fio (Hallinan, 2006)

É sabido que o Linux está sendo amplamente empregado em sistemas embarcados e existe uma tendência de alta em seu uso (Palazzi et al., 2010). Além de não requerer licença para uso, o Linux possui recursos de programação poderosos (Ingle, 2013).

O OpenWRT é um Linux embarcado orientado para suporte a rede (OpenWRT Project, 2015) e fácil de configurar. Além dos recursos supracitados, o OpenWRT foi escolhido para este trabalho, pelas seguintes razões:

- Suporte nativo para linguagens de programação: Bourne shell, Lua e C.
- Suporte nativo para controle de tráfego, incluindo gerência de banda.
- Disponibilidade do comando "iw", poderoso utilitário de linha de comando capaz de coletar dados sobre conexões sem fio tais como taxa de vazão, RSSI, SNR e outros.

- É configurável, possui vários pacotes de *software* opcionais, como por exemplo um agente SNMP.

4.1.2 OpenWRT Atuando como Roteador

O OpenWRT foi projetado especialmente para trabalhar como um roteador. Ele opera um ponto de acesso sem fio, roteando o tráfego originado nas estações cliente para a rede cabeada, e vice-versa. Por sua vez, a rede cabeada roteia o tráfego para a Internet.

Um roteador tem total controle sobre o tráfego que passa por ele. Ele possui *buffers*, onde os datagramas são postos em filas antes que uma decisão seja tomada com relação à qual porta de destino serão encaminhados. Uma tabela de roteamento é consultada para guiar as decisões. Além de fazer roteamento, outros tipos de processamento podem ser aplicados aos pacotes que entram ou saem. A informação contida no cabeçalho dos pacotes TCP/IP podem ser usadas para identificar diferentes tipos de tráfego. Com base neste tipo de informação, pacotes de um determinado tipo podem passar por um processamento diferenciado.

O Controle de tráfego, que é facilmente configurado no OpenWRT (Hubert, 2015; Ke Lin e Ai-qun Hu, 2003; Vila-Carbo et al., 2008; Cano, E. e D. K. Francesco, 2015), é feito usando-se objetos pré-definidos cujos nomes são: “filters", "queues", "qdiscs", "policies" e "classes". "Classes" são conjuntos de "queues". "Filters" são filtros aplicados sobre os pacotes que são postos em fila, de acôrdo com a informação contida em seus cabeçalhos, a fim de selecionar algum tipo de tráfego. Após a classificação do tráfego por intermédio dos "filters", seus pacotes são armazenados em *buffers* à parte chamados de "qdiscs" ou disciplinas de filas. “queues” são filas de pacotes armazenados nos *buffers* do roteadores, antes de serem transmitidos.

"Policies" são aplicadas sobre as "qdiscs", tornando possível, por exemplo:

- Reescalonar ou descartar pacotes indesejáveis.
- Escolher em qual porta entregar os pacotes

- Aplicar atrasos a fim de reduzir a vazão de determinado tipo de tráfego.
- Definir "queues" com prioridade, tais como streamming de voz ou de vídeo.

4.2 Medição de Vazão

Os três scripts usados para medições de taxa de vazão na rede sem fio foram adaptados e reescritos com base nos conceitos expostos por (Marques, 2013). Primeiro serão apresentados dois programas utilitários importantes (Iperf e “iw”) utilizados em conjunto com os scripts de medições.

4.2.1 Iperf

O *software Iperf* é muito utilizado para análise de desempenho da rede e possui várias funcionalidades (Iperf, 2015). O *Iperf* é utilizado para gerar tráfego na rede, no intuito de simular um ambiente real, utilizando a arquitetura de programação distribuída, ou cliente-servidor. O servidor roda no OpenWRT em modo *"daemon".* Modo *"daemon"* significa que o programa não termina nunca, sempre estará rodando enquanto o sistema estiver ativo. Do lado da estação conectada, roda-se o *Iperf* em modo cliente. A estação cliente transmite dados constantemente para o ponto de acesso em sua vazão máxima, através do *Iperf* (Iperf, 2015).

O tráfego de dados em direção à Internet é chamado de *upstream* enquanto que o tráfego de dados originado na Internet e em direção às estações é chamado de *downstream*. Este estudo considera somente o tráfego *upstream*, simulado através desta ferramenta para análise de rede.

4.2.2 O programa utilitário "iw"

O programa utilitário "iw" vem embutido no OpenWRT e serve de apoio à interface de rede sem fio. Através dele, é possível coletar todas as medições de desempenho da rede. Na figura 9, é apresentada uma tela com a saída do comando "iw".

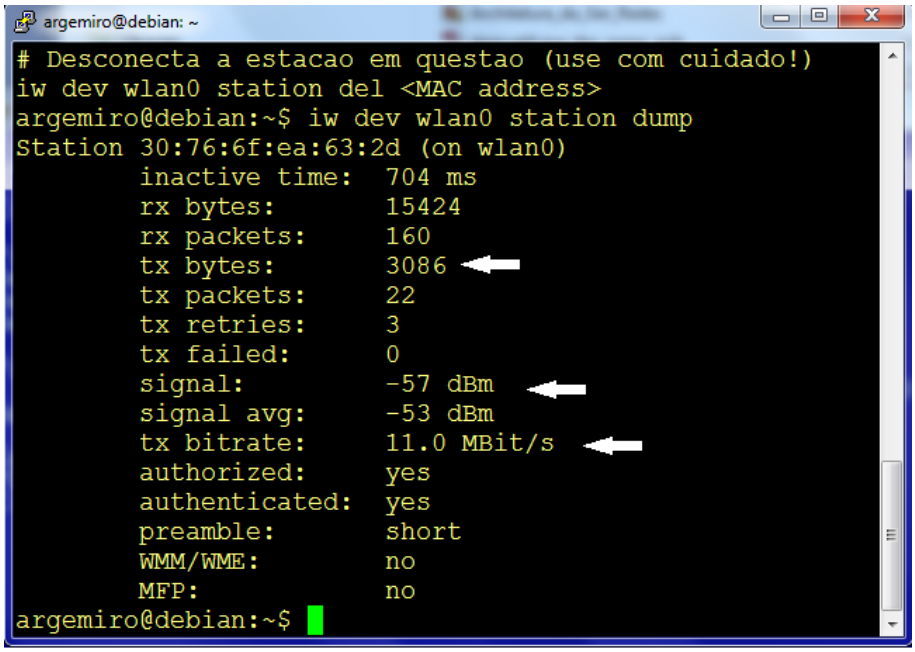


Figura 10: Tela com a saída do comando "iw"

Através do aplicativo “Putty” que é um emulador de terminais “xterm”, conecta-se ao ponto de acesso, obtendo-se uma interface de linha de comando conforme mostrada pela figura 9. Rodando-se o comando “iw dev wlan0 station dump”, por exemplo, serão exibidos todos os dados de desempenho de todas as conexões Wi-Fi naquele instante. Através de programação em Linux Bourne shell, é feito um tratamento destes dados exibidos. Importantes parâmetros de comunicação podem ser coletados como por exemplo: “tx bytes” que é um contador da quantidade de bytes transmitidos desde o início da conexão, até o momento atual. “tx bit rate” é a vazão negociada entre a estação conectada e o ponto de acesso com a qual serão feitas as trocas de pacote. A vazão é negociada automaticamente e constantemente, lenvando-se em conta parâmetros de QoS *(Qualitity of Service)* como “signal” que é a intensidade do sinal de rádio recebido e a SNR *(Signal to Noise Rate).*

4.2.3 Scripts para coleta de dados

Para cada estação conectada ao ponto de acesso, é coletada a taxa de vazão em Mbits/s. É feita uma leitura por segundo e, no final, o programa salva em disco um arquivo de dados com o nome formado pelo MAC *address* da estação.

Tem-se três *scripts*: um somente para listar os endereços MAC de todas estações conectadas; um segundo gerencia as medições de todas as estações e um terceiro faz as medições propriamente ditas, salvando os dados em arquivo. A seguir, nas figuras 10, 11 e 12 são apresentados os fluxogramas de cada *script,* contendo a lógica. No apêndice 8.1 são mostrados os códigos em linguagem Bourne shell.

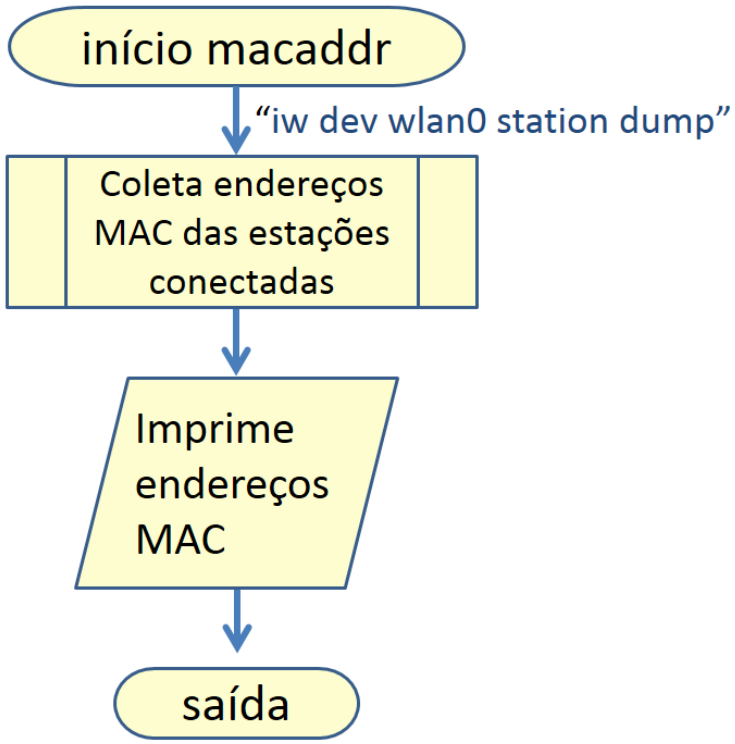


Figura 11: Script que lista endereço MAC das estações

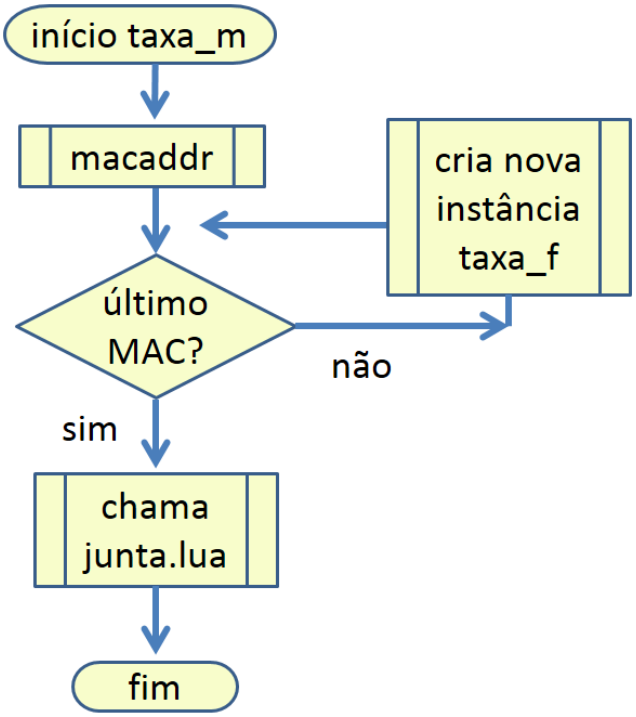


Figura 12: Script principal, inicia e termina as medições

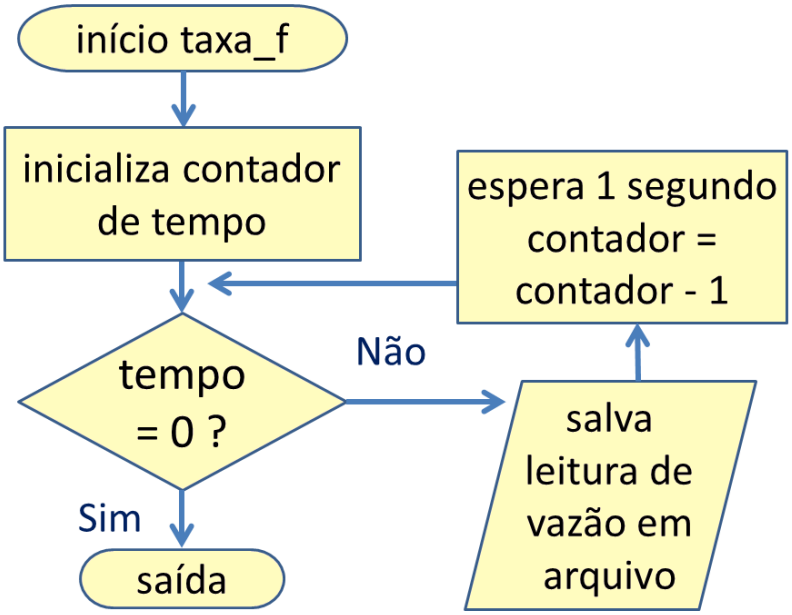


Figura 13: Script que coleta dados de vazão e grava em arquivos

4.3 Identificação de Ofensores

Os arquivos contendo dados de vazão coletados são usados para detectar a presença da anomalia da MAC, bem como para a identificação das estações ofensoras. Os dados de vazão são carregados no ambiente SciLab (SciLab, 2015) e as curvas relacionadas à vazão de cada estação conectada são plotadas. O *software* “SciLab” roda *off-line* em um PC e foi utilizado neste trabalho somente para imprimir os gráficos das medições. Toda a programação em linguagem “Lua” rodando no AP, efetua a identificação *on-line* de eventuais ofensores de maneira integrada com os scripts de medição e mitigação.

A estação ofensora tem uma particularidade: Comparando-se sua curva de vazão com a curva de vazão total da rede, observa-se uma imagem claramente invertida, como se estivesse refletindo a vazão total da rede em um espelho. Este método utiliza-se apenas da inspeção visual dos gráficos. Comparações entre curvas de vazão das estações com a curva de vazão total da rede, apontam se há anomalia ou não, bem como identificam as estações ofensoras.

Se a vazão total da rede aumenta ao mesmo tempo que em certa estação a vazão diminui, ou se enquanto que em certa estação a vazão aumenta causando uma diminuição na vazão total da rede, então pode-se dizer que a anomalia está instalada e que esta é a estação ofensora. A vazão da estação ofensora varia de maneira oposta à variação da vazão total da rede. Este comportamento é descrito em detalhes em (Marques, 2013) e pode ser observado visualmente nas figuras apresentadas no Capítulo 5.

Além de ser um processo manual, o método visual descrito acima é subjetivo. O método proposto neste trabalho baseia-se na similaridade entre duas curvas. Se a curva de uma estação é similar à curva de vazão total da rede, então a estação NÃO É uma ofensora (CCP positivo); caso contrário (CCP negativo) é uma estação ofensora e está causando a anomalia. O método de identificação de ofensores através do cálculo do CCP foi codificado em um *script* em linguagem Lua e instalado no ponto

de acesso. A lógica do programa é apresentada pela figura 13 e o seu código encontra-se no anexo 8.2.

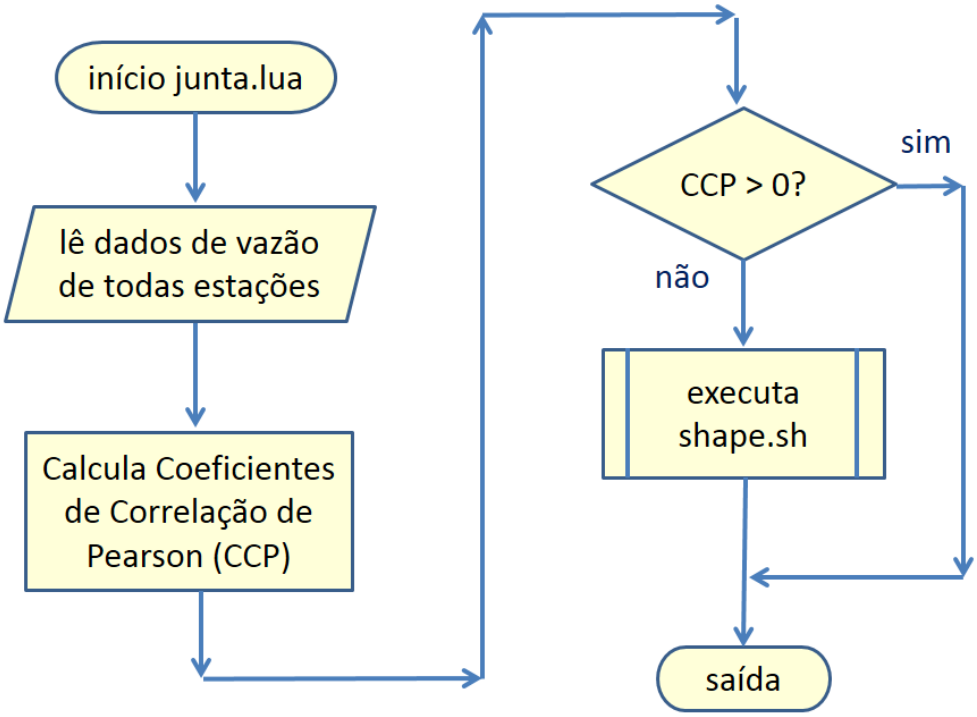


Figura 14: *Script* para Cálculo do CCP

Após o cálculo do CCP para cada uma das estações em relação à vazão total da rede, o *script* junta.lua constrói uma tabela na memória, contendo todos os endereços MAC com os respectivos coeficientes, conforme exemplo da tabela 2 a seguir. Em seguida, todos os *traffic shapes* que porventura existam, são removidos e para cada valor negativo da coluna “coeficiente”, será criado um novo *traffic shaping* chamando o *script* shape.sh, passando-se como argumento o endereço MAC.

Tabela 3: Coeficientes de Correlação de Pearson

Rodada junta.lua	Endereço MAC	Coeficiente
1	00-21-5c-06-21-4f 08-11-96-de-a1-98 70-f1-a1-d1-66-ad e0-ca-94-31-75-05	-0.78545179221401 0.75718225413704 0.44069650229212 0.77148623490922
2	00-21-5c-06-21-4f 08-11-96-de-a1-98 50-b7-c3-c1-21-b8 70-f1-a1-d1-66-ad e0-ca-94-31-75-05	-0.3383039902961 0.45133774244948 0.019827578030685 0.5558939481801 0.48939326066343
3	00-21-5c-06-21-4f 08-11-96-de-a1-98 70-f1-a1-d1-66-ad e0-ca-94-31-75-05	0.58655971625961 -0.12571298792616 0.77543694051413 -0.15778394633392

4.4 Mitigação, o Script para *Traffic Shaping*

Quanto maior for o tráfego gerado pela estação ofensora, mais marcante será a anomalia na MAC. *Traffic shaping* é o método empregado neste trabalho para mitigar a Anomalia da MAC, objetivando restaurar a operação normal da rede. Vale ressaltar que este método já foi utilizado para esta finalidade, conforme atesta a referência (Garropo, 2006). Basicamente, este método consiste em limitar a vazão das estações ofensoras, ocasionando assim um aumentando na vazão total da rede, atingindo-se o mesmo nível anterior à instalação da Anomalia. A estratégia adotada é forçar uma redução na vazão da estação ofensora. Assim, esta não afetará o restante da rede, ainda tendo oportunidade para transmitir dados. Em quanto deve-se reduzir esta vazão e qual o valor de corte mínimo? Neste trabalho foi adotado o valor fixo de 2Mbits/s para aplicação do *traffic shaping* e nenhum cálculo foi feito neste sentido. Este valor foi o mais baixo possível, somente para que ainda seja dada oportunidade de a estação continuar transmitindo. É possivel de se fazer um ajuste fino tanto no *script* que faz a identificação de ofensores quanto no *script* que faz o *traffic shaping*. No script “junta.lua”, é possível desprezar ofensores que apresentem um baixo grau de ofensividade, como por exemplo, Coeficiente de Correlação de Pearson > -0.3.

Assim, estes ofensores seriam tolerados e nenhuma ação seria tomada. Quanto ao *script* “shaping.sh”, poderia variar-se o valor do *shaping* adotando-se valores mais apropriados em vez do valor fixo em 2Mbits/s. Uma sugestão seria a metade da vazão efetiva praticada pela estação ofensora. Neste trabalho não foram testados estes resultados.

Uma explicação completa do processo de *traffic shaping* pode ser encontrada em (Hubert, 2015), que é um tutorial completo sobre controle de tráfego em Linux.

Um *script* pequeno e simples que implementa *traffic shaping* foi desenvolvido. Seu fluxograma é apresentado na figura 14. Seu código é apresentado no anexo 8.3 e sua lógica é explicada a seguir:

1) Antes de mais nada, é necessário ter o MAC *address* da estação ofensora para que seja passado como argumento de linha de comando no carregamento do *script*. É utilizado o endereço MAC identificado utilizando-se o método descrito na seção 3.2.
2) Criar uma tabela usando o comando "iptables". Esta tabela vai capturar todos os pacotes entrantes e marcá-los, caso seu MAC *address* seja igual àquele passado como argumento na linha de comando.
3) Criar uma *"ingress queue discipline"* ou "qdisc". "qdisc" é uma fila para armazenar os pacotes do tráfego entrante.
4) Criar um filtro e aplicá-lo sobre a "qdisc". A ligação entre o filtro e a "qdisc" é feita usando-se a cláusula "parent" que contém o código de identificação da "qdisc" "ffff:".O filtro selecionará apenas os pacotes marcados pela tabela criada por "iptables", isto é originados pela estação ofensora. A ligação entre o filtro e a tabela é especificada pela cláusula "handle 1".
5) Aplicar uma política nestes pacotes. A política fará o seguinte: Ponha todos os pacotes em um *buffer* de tamanho 10k bytes. Retransmita-os apenas se este buffer não estiver cheio, a uma taxa de 2Mbits/s. Se o *buffer* já estiver cheio e mais pacotes chegarem, então estes pacotes serão descartados. Os pacotes que chegaram e foram descartados não serão perdidos. Após o não recebimento de um ACK durante um certo período de tempo acusando *time-out,* o nó transmissor

baixará sua taxa de transmissão e retransmitirá todos os pacotes descartados. Assim, conseguimos forçar uma queda na taxa de vazão da estação ofensora.

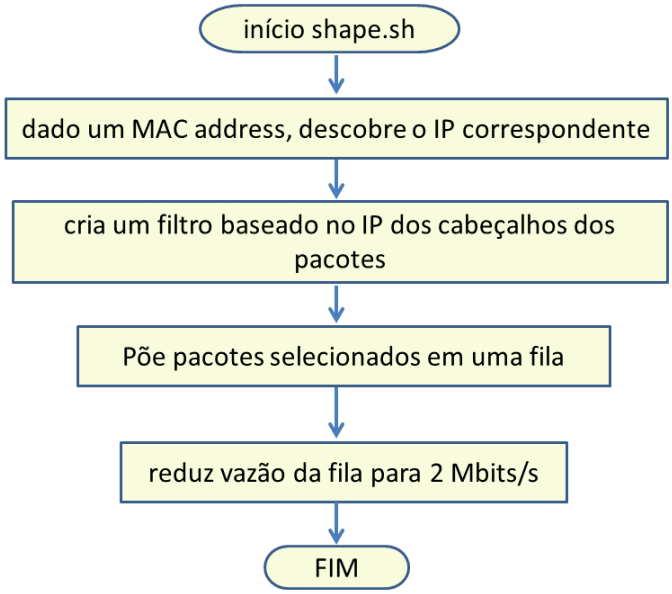


Figura 15: Script para *Traffic Shaping*

4.5 Putty, WinSCP e SciLab

Outras ferramentas de *software* utilizadas foram: Putty, WinSCP e SciLab.

Putty: Rodando em modo ssh, Putty foi usado para iniciar e parar os scripts de medição, identificação de ofensores e *traffic shaping,* que rodam integrados. Também foi usado Putty para fazer todas as instalações e configurações no OpenWRT.

WinSCP: Os resultados foram salvos no próprio OpenWRT, mas devido à sua memória de tamanho reduzido e volátil, o WinSCP foi utilizado para transferir os dados para posterior processamento no SciLab.

SciLab: Foi usado para compilar os dados de todas as estações e construir os gráficos usando a sua própria linguagem de programação. O SciLab também foi usado inicialmente para programação do *script* que faz o cálculo do coeficiente de Pearson, antes da criação do programa definitivo em linguagem Lua, para ser rodado no OpenWRT.

5. RESULTADOS E DISCUSSÕES

Os *scripts* para medição de vazão, identificação de ofensores e mitigação da anomalia foram integrados e rodam em tempo real de maneira automática, sequencialmente. A bancada pode ser levada a campo para testes e medições com equipamentos convencionais utilizados para acesso à Internet sem fio. Os resultados das medições serão apresentados a seguir.

5.1 Testes de Desempenho da Bancada

O processamento mais pesado do ponto de vista computacional, é com o *script* que faz a identificação dos ofensores. Após os dados de vazão de cada estação serem salvos em arquivos pelo *script* que fez as medições, é chamado o *script* que faz a identificação dos ofensores. Este script junta tudo em uma grande tabela para que sejam feitos os cálculos do coeficiente de correlação de Pearson (junta.lua). Cada linha desta tabela contém as leituras de vazão de até 15 estações, feitas naquele segundo. Portanto, o número de linhas da tabela equivale ao tempo gasto em segundos para fazer as medições. A figura 15 mostra o tempo gasto em processamento em função do número de linhas processadas. Foram criados 15 arquivos em disco, simulando a conexão de 15 estações. Em seguida foram sendo adicionadas mais linhas de dados à estes arquivos simulando leituras de dados. Conforme foram sendo aumentadas as linhas nos arquivos, uma nova rodada do script “junta.lua” foi feita, medindo-se o seu tempo de execução. Observa-se uma linearidade no gráfico da figura 15. O tempo de processamento é diretamente proporcional ao número de linhas. Portanto, o ponto de acesso levou 1s para processar 15 arquivos contendo 500 linhas cada um, equivalentes a 500s, de medição.

Para melhorar desempenho, pode-se utilizar outros equipamentos com maior capacidade de processamento e reescrever os programas na linguagem C, compilando-os em seguida.

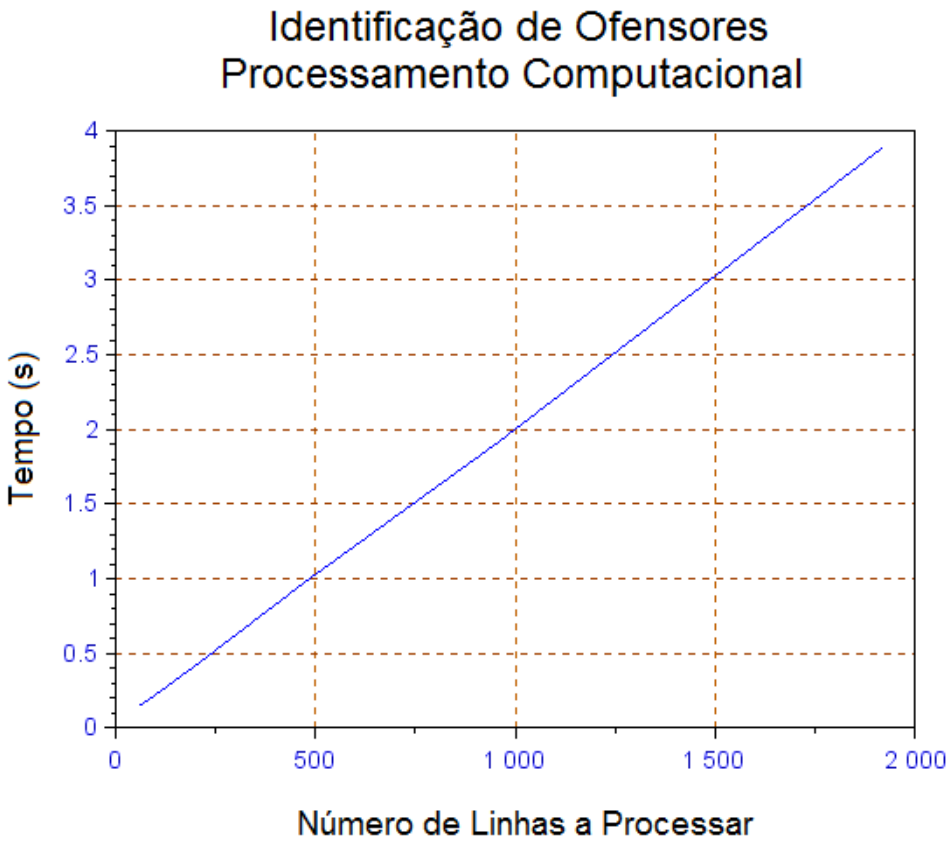


Figura 16: Desempenho do Script de Identificação de Ofensores

5.2 Testes de Medição de Vazão

É disparada uma instância do programa de leitura de vazão para cada estação conectada. Além disso, é disparada uma instância do programa *Iperf* para cada estação conectada em modo servidor no ponto de acesso. Ao todo, são gastos 25% do tempo da CPU, o que prejudica a própria vazão total da rede pois a CPU está compartilhando o seu tempo de operação da rede.

O tempo ideal em que o programa de medição de vazão deverá permanecer rodando tem de ser estimado. Foi atribuído um tempo em torno de 10 minutos para cada ciclo abrangendo medição, detecção e mitigação.

Após o término da execução das medições, são salvos arquivos em disco que são usados pelo programa que faz a identificação de possíveis ofensores. Os arquivos em disco também são usados no SciLab para montagem dos gráficos.

5.3 Testes de Identificação de Ofensores

Conforme já exposto nas seções 3.2 e 4.3, foram adotados dois métodos para a identificação de ofensores: inspeção visual ou CCP. O método da inspeção visual é apresentado aqui somente para ilustração e foi usado para confirmação do método do CCP. Neste trabalho foram realizados testes de identificação de ofensores em cenários contendo 2, 3, 4 ou 5 estações. Visualmente, pode-se identificar ofensores claramente num cenário com até 3 estações; a partir daí vai ficando mais difícil de se visualizar o(s) ofensor(es). Os cenários utilizados para os testes encontram-se descritos a seguir:

5.3.1 Cenário 1 – Anomalia com 2 estações conectadas

A estação “A” não ofensora, representada pela curva em azul nos gráficos, possui as seguintes características:

Modelo: *lap top* RV410
Fabricante: SAMSUNG Electronics
Rede Wi-Fi: Chip Atheros AR9285 Wireless LAN Adapter 802.11n até 150 Mbps
Processador: Intel Celeron Dual Core T350D, clock de 2.10GHz, 2Gb memória.
Sistema: Windows 7 Starter SP1 32 bits
Taxas máximas medidas: TCP 45Mbits/s, UDP 58Mbits/s
Data de fabricação: março/2011

A estação “B” ofensora, representada pela curva em preto nos gráficos, possui as seguintes características:

Modelo: *lap top* 6910p
Fabricante: Hewlett Packard do Brasil Ltda.
Rede Wi-Fi: Intel Wireless Link 4965 a/g/n
Processador: Intel Core 2 Duo T8100 2.10 GHz
Sistema: Windows 7 Ultimate 64 bits
Taxas máximas medidas: TCP 47Mbits/s, UDP 63.3 Mbits/s
Data de fabricacao: janeiro/2008

Para que fosse simulado um ambiente real de uso, foram rodadas duas instâncias do software “Iperf” em modo servidor no ponto de acesso: uma instância na porta 5001 e outra na porta 5002. Nos lap-tops, o “Iperf” foi rodado em modo cliente, cada um conectando-se a uma das referidas portas, operando sempre em sua vazão máxima, transmitindo dados ao ponto de acesso utilizando o protocolo TCP.

A estação “B” ofensora, foi sendo distanciada do ponto de acesso até que fosse instalada a anomalia. A anomalia ocorreu a uma distância de 10m em ambiente construido, tendo duas paredes atenuando o sinal. A estação “A” não ofensora, permaneceu a uma distância de 1,5m do ponto de acesso.

A figura 16 apresenta as medições de vazão das estações A e B, mais a vazão total da rede, durante 93s. A figura 17 é um destaque comparando-se a vazão da estação “A” não ofensora, com a vazão total da rede. Nota-se claramente uma similaridade nos gráficos, confirmada pelo CCP ρ=0.9153457, indicando uma correlação acentuada positiva. Na figura 19, é destacada a estação “B”, ofensora. Nota-se claramente que o gráfico de vazão da estação “B” tem um comportamento inverso ao do gráfico da vazão total da rede. Quando sobe o nível de vazão da estação “B”, desce o nível da vazão total da rede, e vice-versa. Este comportamento invertido, caracteriza a estação como ofensora. A ofensividade é confirmada pelo CCP negativo ρ = -0.4729942.

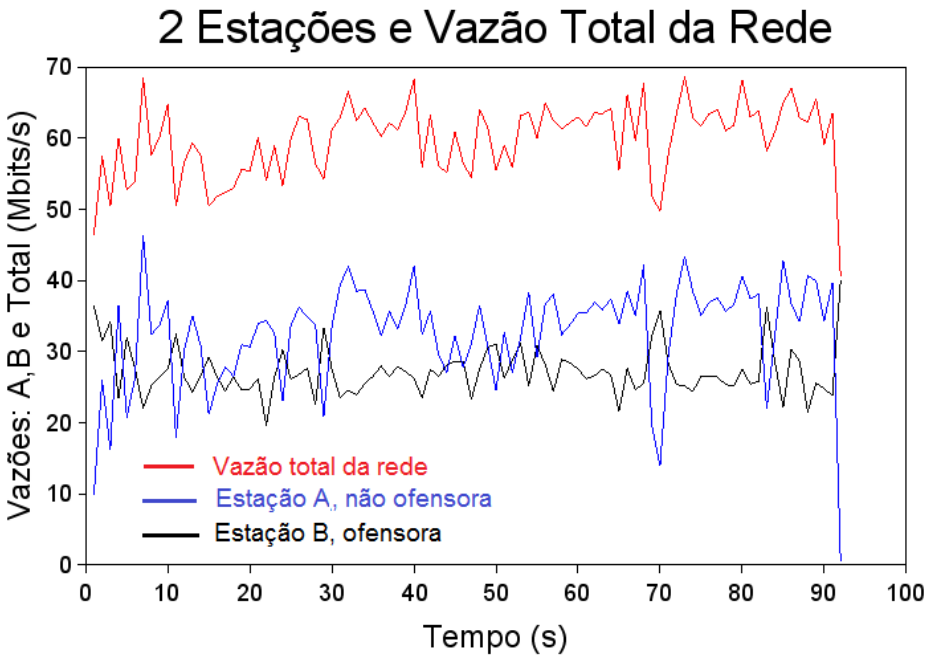


Figura 17: Cenário 1 - Anomalia com 2 estações

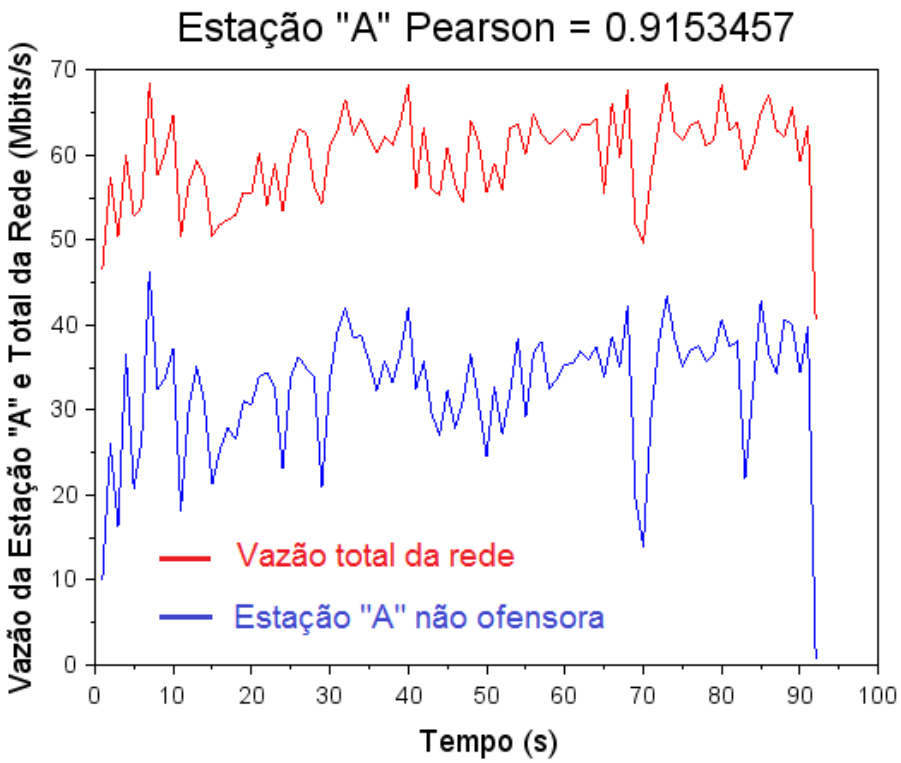


Figura 18: Cenário 1 - Destaque da estação “A”, não ofensora

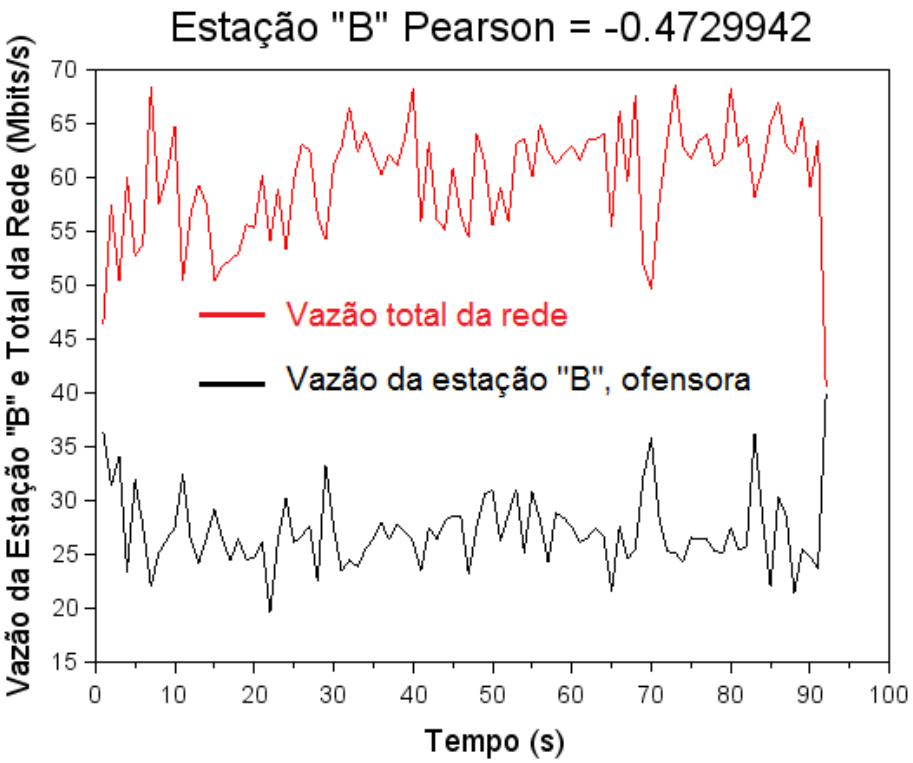


Figura 19: Cenário 1 - Destaque da estação “B”, ofensora

5.3.2 Cenário 2 – Anomalia com 3 estações conectadas

A estação “A” não ofensora, representada pela curva em azul nos gráficos, possui as seguintes características:

Modelo: *lap top* RV410
Fabricante: SAMSUNG Electronics
Rede Wi-Fi: Chip Atheros AR9285 Wireless LAN Adapter 802.11n até 150 Mbps
Processador: Intel Celeron Dual Core T350D, clock de 2.10GHz, 2Gb memória.
Sistema: Windows 7 Starter SP1 32 bits
Taxas máximas medidas: TCP 45Mbits/s, UDP 58Mbits/s
Data de fabricação: março/2011

A estação “B” não ofensora, representada pela curva em preto nos gráficos, possui as seguintes características:

Modelo: ProBook-6460b
Fabricante: Hewlett Packard do Brasil Ltda
Wi-Fi: Padrao 802.11n Intel Centrino Advanced N 6205
Processador: Intel i5-2520M 2.5GHz
Sistema: Windows 7 Enterprise SP1 64bits
Taxas máximas medidas: TCP 48 Mbits/s, UDP 66.8 Mbits/s
Data de fabricação: junho/2011

A estação “C” ofensora, representada pela curva em verde nos gráficos, possui as seguintes características:

Modelo: *lap top* 6910p
Fabricante: Hewlett Packard do Brasil Ltda.
Rede Wi-Fi: Intel Wireless Link 4965 a/g/n
Processador: Intel Core 2 Duo T8100 2.10 GHz
Sistema: Linux Ubuntu (dual boot)
Taxas máximas medidas: TCP 47 Mbits/s, UDP 63.3 Mbits/s
Data de fabricacao: janeiro/2008

Para que fosse simulado um ambiente real de uso, foram rodadas três instâncias do software “Iperf” em modo servidor no ponto de acesso, alocando as portas 5001, 5002 e 5003. Nos lap-tops, o “Iperf” foi rodado em modo cliente, cada um conectando-se a uma das referidas portas, operando sempre em sua vazão máxima, transmitindo dados ao ponto de acesso utilizando o protocolo TCP.

As três estações permaneceram próximas ao ponto de acesso a uma distância aproximada de 2m. Para que ocorresse a anomalia, a estação ofensora foi forçada a operar a uma vazão mais baixa que as demais: 8Mbits/s. Para tanto, utilizou-se o comando “iw” disponível no Linux.

A figura 21 exibe as medições de vazão das três estações: A, B e C; mais a vazão total da rede, durante 200s. As figuras 22 e 23 são um destaque exibindo as duas estações não ofensoras A e B. Comparando-se visualmente as vazões das

estações não ofensoras com a vazão total da rede, observa-se claramente uma similaridade, confirmada pelo cálculo do CCP: estação “A” ρ=0.9339384 e estação “B” ρ=0.9215047. A figura 24 é um destaque para a estação “C”, ofensora. Comparando-se visualmente a sua curva de vazão com curva de vazão total da rede, observa-se claramente um gráfico invertido, confirmado pelo cálculo do CCP para a estação “C”: ρ=-0.9054236.

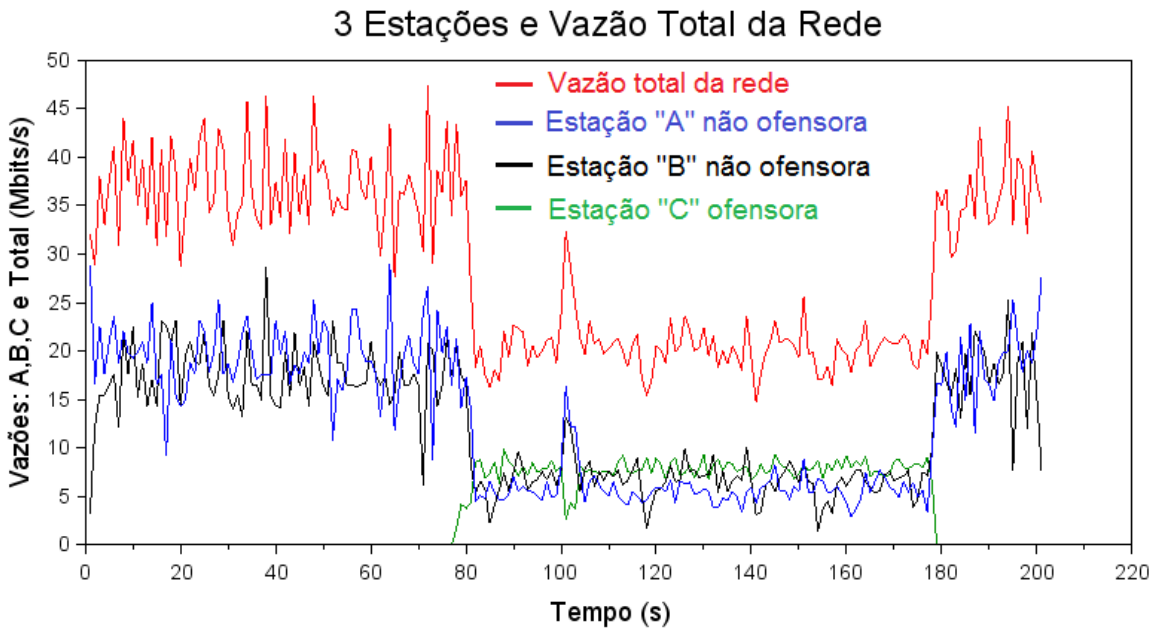


Figura 20: Cenário 2 - Anomalia com 3 estações, 1 ofensora

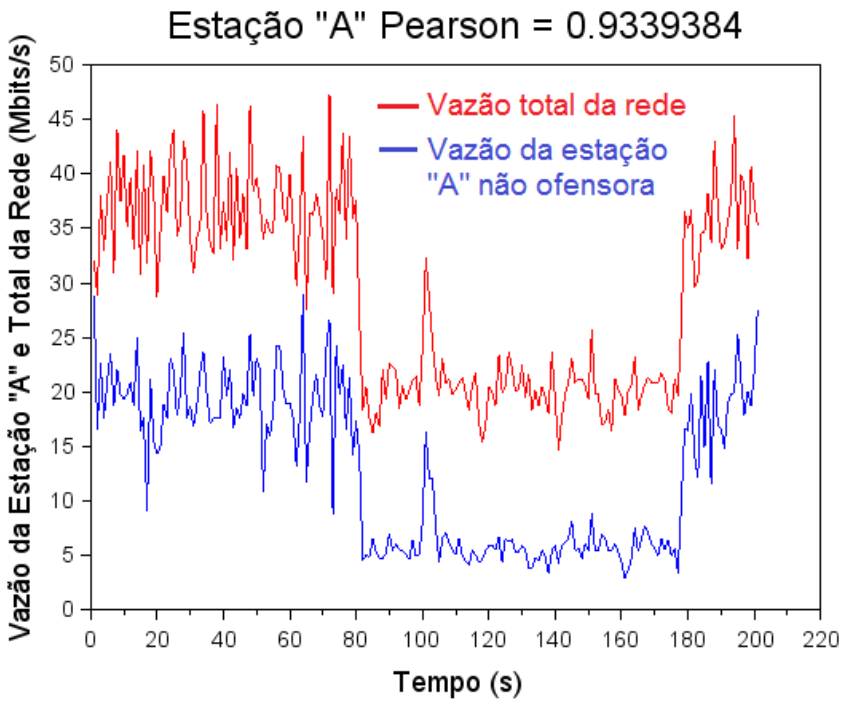
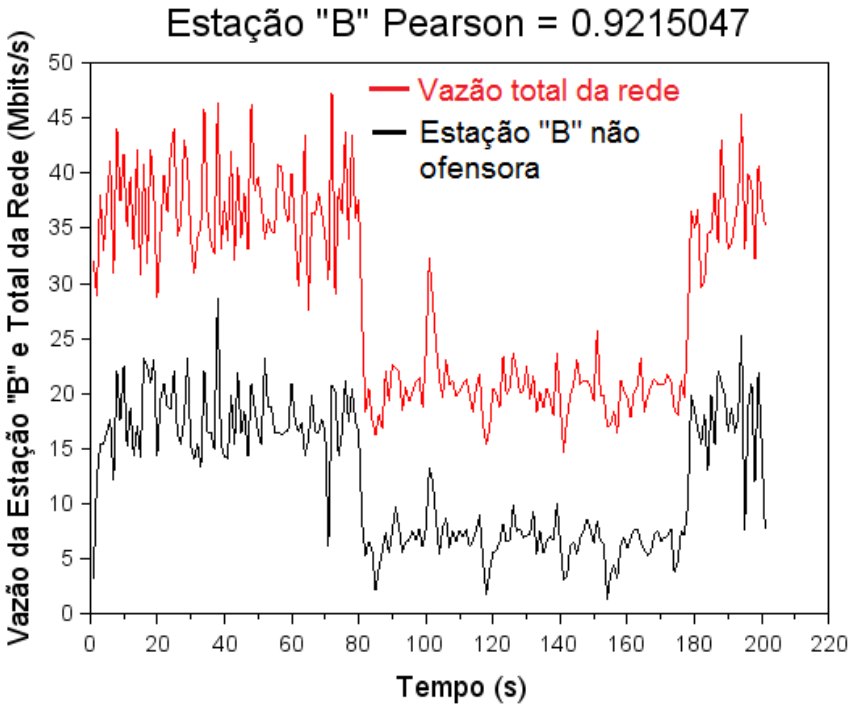


Figura 21: Cenário 2 - Destaque da estação “A”, não ofensora



Figuras 22: Cenário 2 - Destaque da estação “B” não ofensora

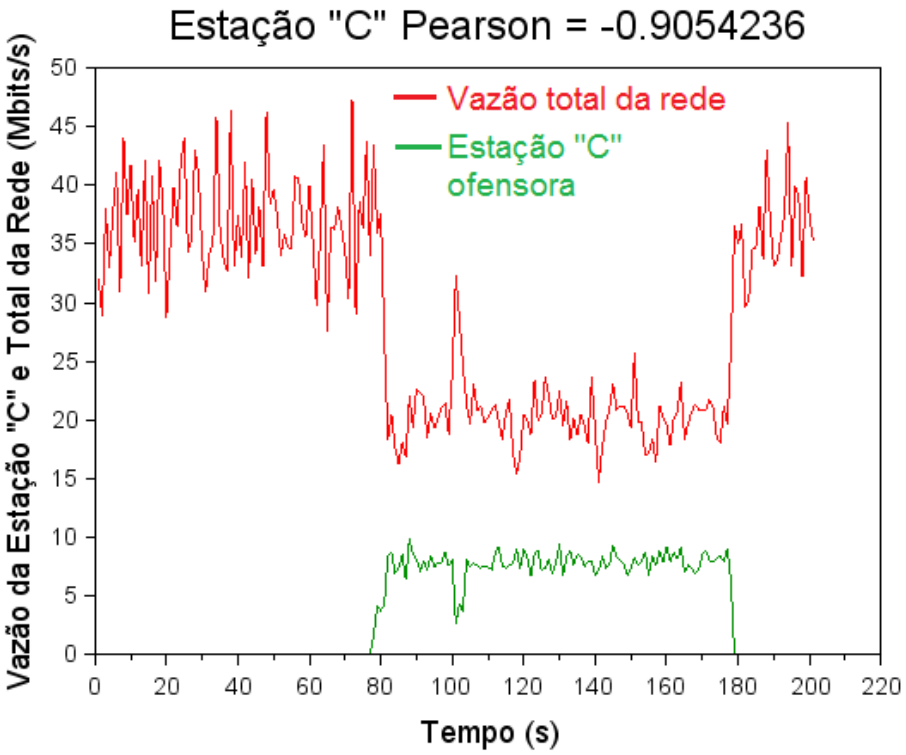


Figura 23: Cenário 2 - Destaque da estação “C”, ofensora

5.3.3 Cenário 3 - Anomalia com 4 estações conectadas

A estação “A” não ofensora, representada pela curva em côr azul nos gráficos, possui as seguintes características:

Modelo: *lap top* RV410
Fabricante: SAMSUNG Electronics
Rede Wi-Fi: Chip Atheros AR9285 Wireless LAN Adapter 802.11n até 150 Mbps
Processador: Intel Celeron Dual Core T350D, clock de 2.10GHz, 2Gb memória.
Sistema: Windows 7 Starter SP1 32 bits
Taxas máximas medidas: TCP 45Mbits/s, UDP 58Mbits/s
Data de fabricação: março/2011

A estação “B” não ofensora, representada pela curva em côr preta nos gráficos, possui as seguintes características:

Modelo: ProBook-6460b
Fabricante: Hewlett Packard do Brasil Ltda
Wi-Fi: Padrõo 802.11n Intel Centrino Advanced N 6205

Processador: Intel i5-2520M 2.5GHz
Sistema: Windows 7 Enterprise SP1 64bits
Taxas máximas medidas: TCP 48 Mbits/s, UDP 66.8 Mbits/s.
Data de fabricação: junho/2011.

A estação “C” não ofensora, representada pela curva em côr verde nos gráficos, possui as seguintes características:

Modelo: *lap top* G42 *Notebook* PC
Fabricante: Hewlett Packard do Brasil Ltda.
Rede Wi-Fi: IEEE 802.11 a/b/g/n
Processador: Pentium Dual Core T4500 2.3GHz
Sistema: Windows 7 *Home Edition* 32 bits
Taxas máximas medidas: TCP 41 Mbits/s, UDP 56 Mbits/s.
Data de fabricação: 2010.

A estação “D” ofensora, representada pela curva em côr rosa nos gráficos, possui as seguintes características:

Modelo: *lap top* 6910p
Fabricante: Hewlett Packard do Brasil Ltda.
Rede Wi-Fi: Intel Wireless Link 4965 a/g/n
Processador: Intel Core 2 Duo T8100 2.10 GHz
Sistema: Linux Ubuntu (dual boot)
Taxas máximas medidas: TCP 47 Mbits/s, UDP 63.3 Mbits/s
Data de fabricação: janeiro/2008.

Para que fosse simulado um ambiente real de uso, foram rodadas quatro instâncias do software “Iperf” em modo servidor no ponto de acesso, alocando as portas 5001, 5002, 5003 e 5004. Nos *lap-tops*, o “Iperf” foi rodado em modo cliente, cada um conectando-se a uma das referidas portas, operando sempre em sua vazão máxima e transmitindo dados ao ponto de acesso utilizando o protocolo TCP.

As quatro estações permaneceram próximas ao ponto de acesso a uma distância aproximada de 2m. Para que ocorresse a anomalia, a estação ofensora foi forçada a operar a uma vazão mais baixa do que as demais: 8Mbits/s. Para tanto, utilizou-se o comando “iw” disponível no Linux.

A figura 26 exibe as medições coletadas das estações A, B, C e D mais a vação total da rede, durante 800s. As figuras 27, 28 e 29 são um destaque para as três

estações não ofensoras. Nota-se uma similaridade das curvas de vazão com a curva de vazão total da rede, confirmadas pelo CCP: estação “A” ρ = 0,7714862, estação “B” ρ = 0,7571823 e estação “C” ρ = 0,4406965. A figura 30 é um destaque para a estação “D” ofensora. Nota-se que a curva de vazão da estação “D”, comparada com a curva da vazão total da rede, é uma imagem invertida, confirmada pelo CCP ρ = -0.7854518.

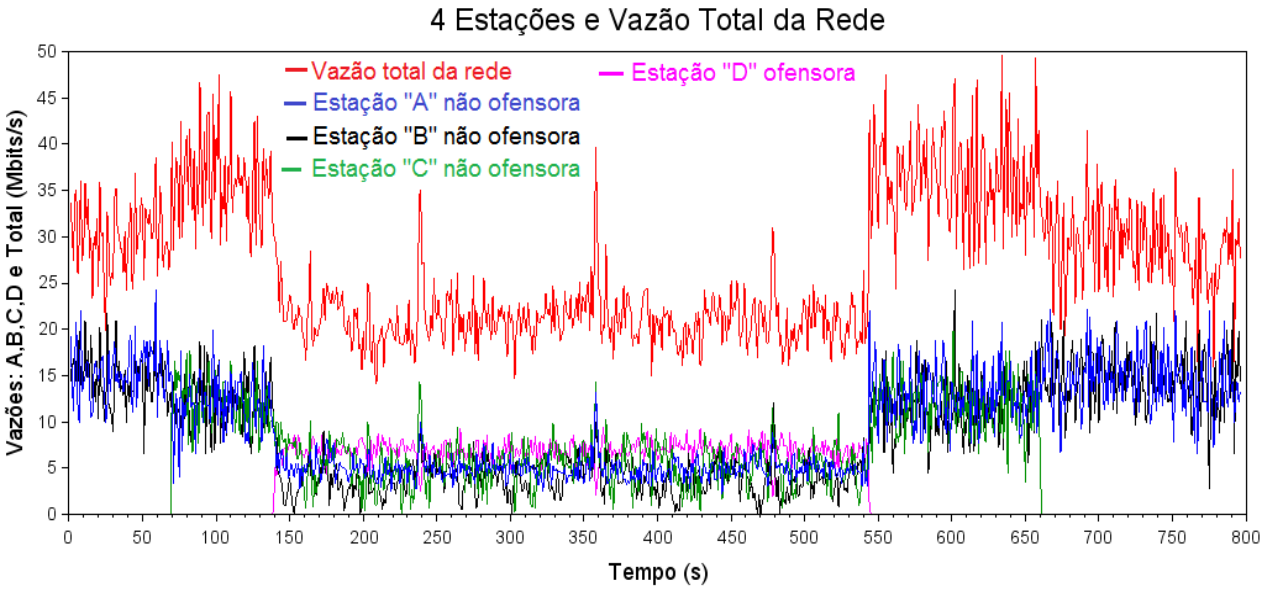


Figura 24: Cenário 3 - Anomalia com 4 Estações

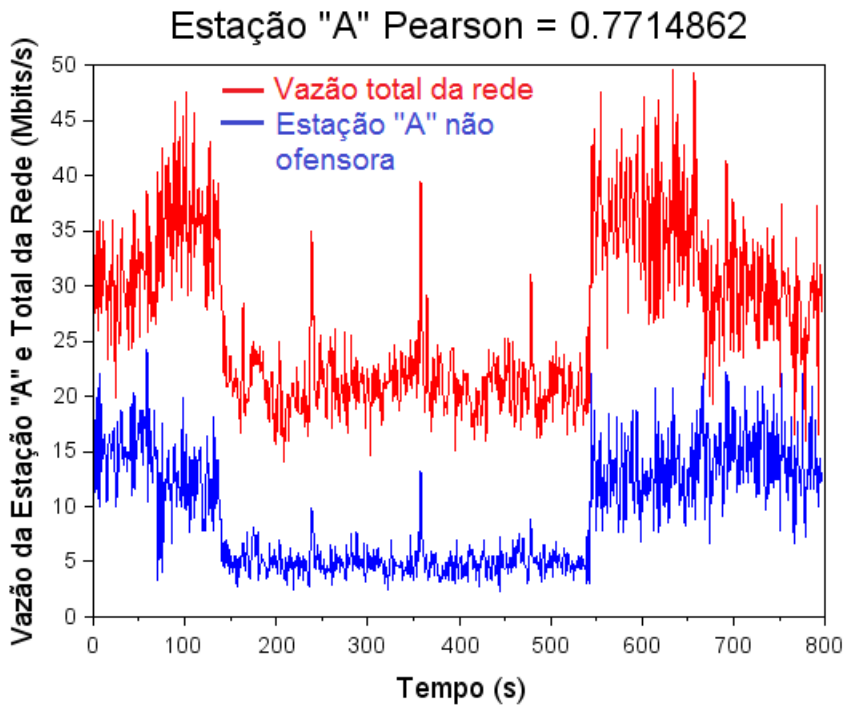


Figura 25: Cenário 3 - Destaque da estação “A”, não ofensora

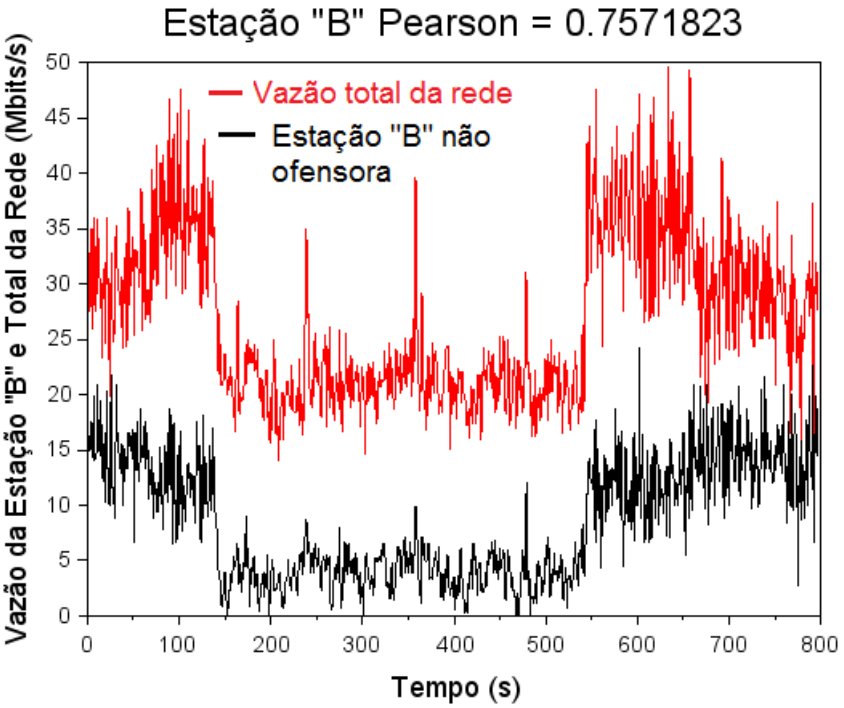


Figura 26: Cenário 3 - Destaque da estação “B”, não ofensora

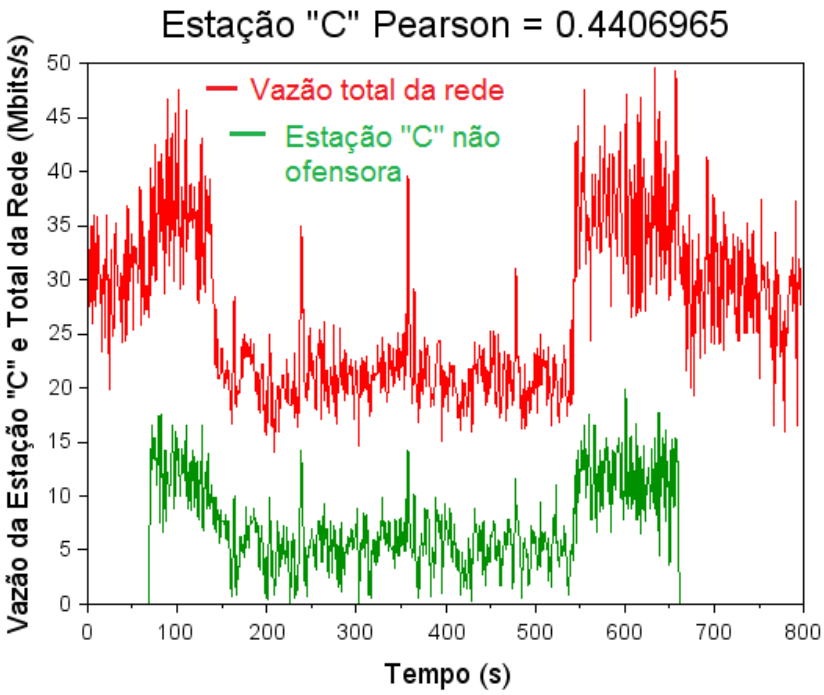


Figura 27: Cenário 3 - Destaque da estação “C”, não ofensora

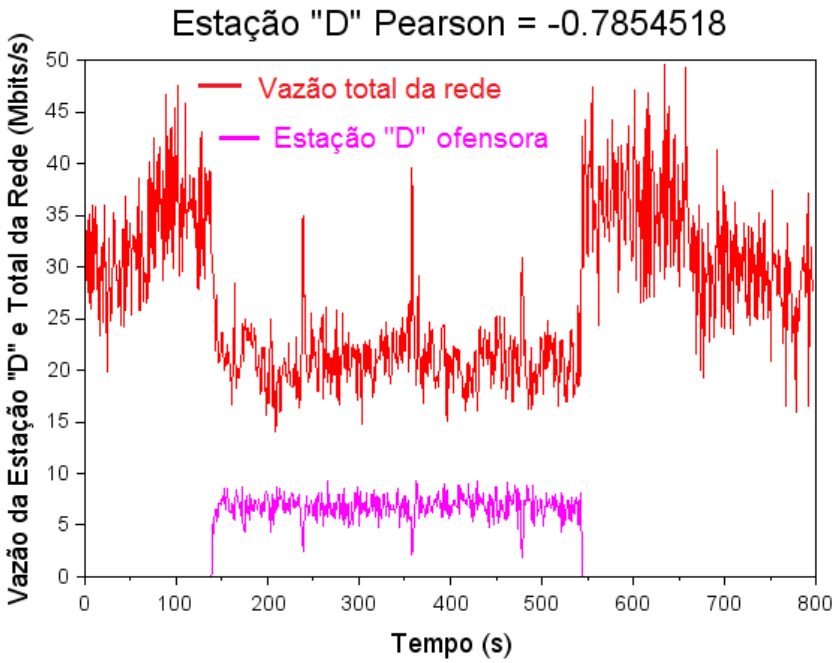


Figura 28: Cenários 3 - Destaque da estação “D”, ofensora.

5.3.4 Cenário 4 – Anomalia com 5 estações conectadas

É mostrado um cenário com anomalia tendo 5 estações conectadas, sendo que 2 delas são ofensoras, no conjunto de figuras 31-36. O método de cálculo do CCP mostrou-se eficiente em cenários com 5 estações conectadas. Conseguiu-se identificar mais de um ofensor, no mesmo cenário. Considerando-se que o tráfego foi simulado com o uso do programa para análise de rede “Iperf”, que gerou vazão máxima para todas as estações, a vazão total da rede deveria se manter em nível máximo também, ou seja em torno de 60 Mbits/s. Mas isto não ocorreu, percebe-se quedas acentuadas no nível de vazão total da rede, devido à presença da anomalia da MAC. Neste cenário com 5 estações, fica difícil de se identificar ofensores apenas com o método de inspeção visual. É necessário construir um gráfico em separado para cada uma das estações, comparando sua vazão com a vazão total da rede. O método de cálculo do coeficiente de correlação, permitiu a identificação dos ofensores com segurança e consequente disparo do programa que faz a mitigação.

A estação “A” não ofensora, representada pela curva em côr azul nos gráficos, possui as seguintes características:

Modelo: *lap top* RV410
Fabricante: SAMSUNG Electronics
Rede Wi-Fi: Chip Atheros AR9285 Wireless LAN Adapter 802.11n até 150 Mbps
Processador: Intel Celeron Dual Core T350D, clock de 2.10GHz, 2Gb memória.
Sistema: Windows 7 Starter SP1 32 bits
Taxas máximas medidas: TCP 45Mbits/s, UDP 58Mbits/s
Data de fabricação: março/2011

A estação “B” não ofensora, representada pela curva em côr preta nos gráficos, possui as seguintes características:

Modelo: *lap top* 6910p
Fabricante: Hewlett Packard do Brasil Ltda.
Rede Wi-Fi: Intel Wireless Link 4965 a/g/n
Processador: Intel Core 2 Duo T8100 2.10 GHz
Sistema: Linux Ubuntu (dual boot)
Taxas máximas medidas: TCP 47 Mbits/s, UDP 63.3 Mbits/s
Data de fabricação: janeiro/2008.

A estação “C” não ofensora, representada pela curva em côr verde nos gráficos, possui as seguintes características:

Modelo: *lap top* ProBook-6460b.
Fabricante: Hewlett Packard do Brasil Ltda.
Wi-Fi: Padrão IEEE 802.11n Intel Centrino Advanced N 6205.
Processador: Intel i5-2520M 2.5GHz.
Sistema: Windows 7 Enterprise SP1 64bits.
Taxas máximas medidas: TCP 48 Mbits/s, UDP 66.8 Mbits/s.

A estação “D” ofensora, representada pela curva em côr rosa nos gráficos, possui as seguintes características:

Modelo: *Smart Phone* HP Slate 7 Voice Tab (HSTNH-B19C)
Fabricante: Hewlett Packard do Brasil Ltda.
Wi-Fi: padrão 802.11 b/g/n até 150 Mbits/s
Sistema: Android 4.4.2, kernel 3.4.39 17/10/2014
Taxa maxima medida: TCP: 43 Mbits/s UDP: 67 Mbits/s
Data de Fabricação: dezembro/2014

A estação ofensora “E”, representada pela curva em côr azul claro nos gráficos, possui as seguintes características:

Modelo: *lap top* 300E4C
Fabricante: SANSUMG Electronics
Wi-Fi: padrão 802.11 a/b/g/n até 150 Mbits/s
Sistema: Windows 8 64 bits
Data de Fabricacao: março/2013
Taxas máximas medidas: TCP 39 Mbits/s, UDP 65 Mbits/s

Para provocar a anomalia, as estações ofensoras “D” e “E” foram afastadas do ponto de acesso até uma distância de 15m em ambiente construído com 3 paredes de alvenaria atenuando o sinal; enquanto que as estações não ofensoras “A”, “B” e “C” permaneceram próximas do ponto de acesso a uma distância de 2m.

A figura 31 exibe as medições de vazão das cinco estações: A, B, C, D e E; mais a vazão total da rede durante 180s. As figuras 32-34 exibem as curvas de vazão das estações não ofensoras A, B e C comparadas com a vazão total da rede, enquanto que as figuras 35 e 36 exibem gráfico similar das estações ofensoras.

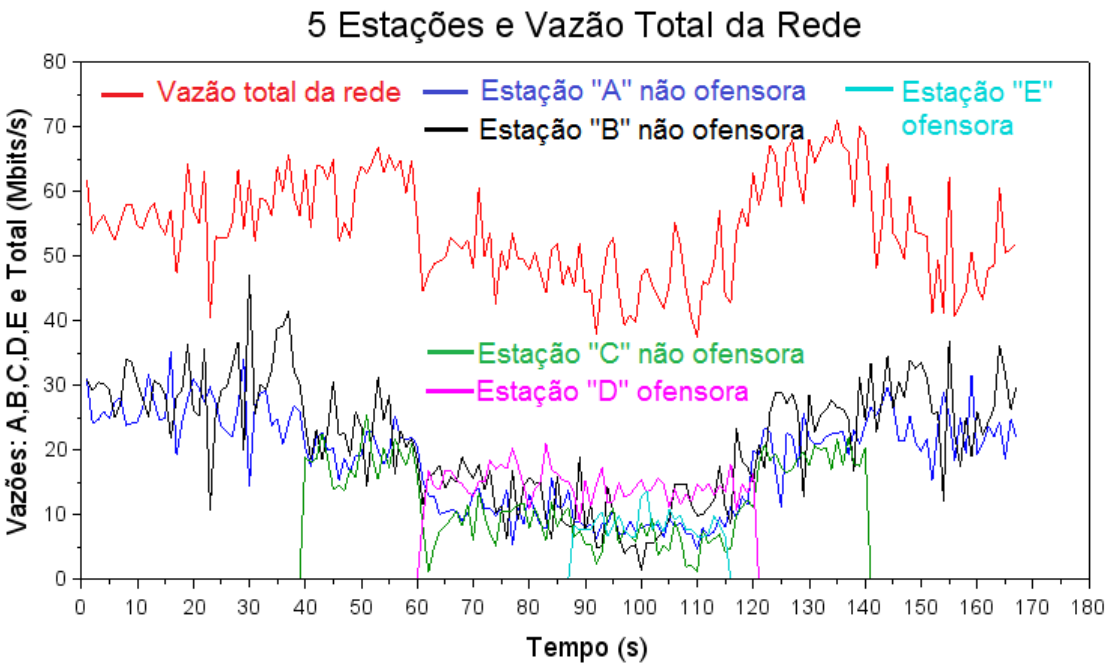


Figura 29: Cenário 4 - Anomalia com 5 estações, 2 Ofensoras

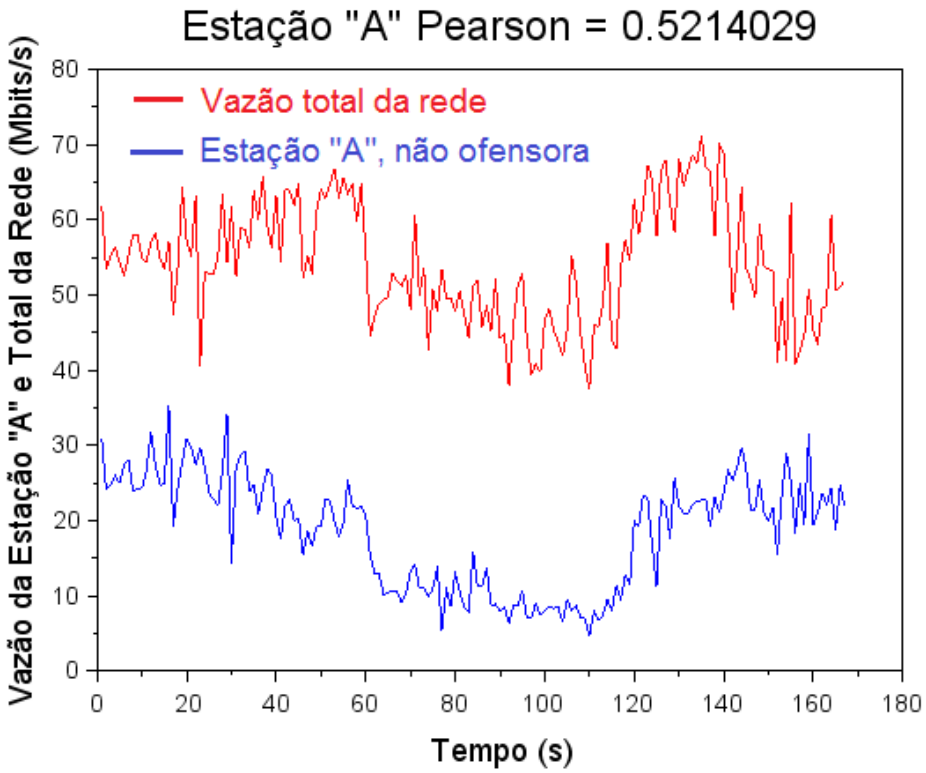


Figura 30: Cenário 4 - Destaque da estação “A”, não ofensora

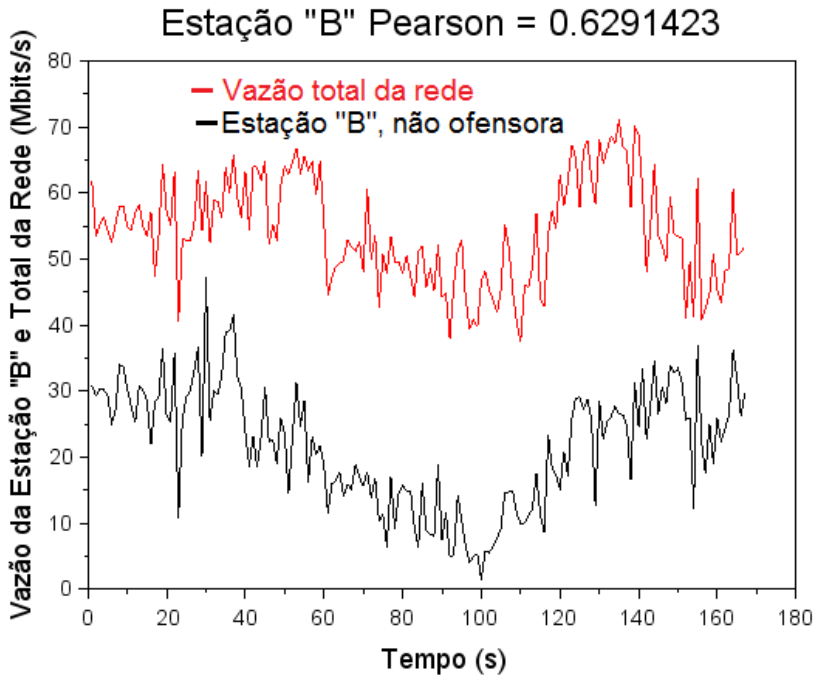


Figura 31: Cenário 4 - Destaque da estação “B”, não ofensora

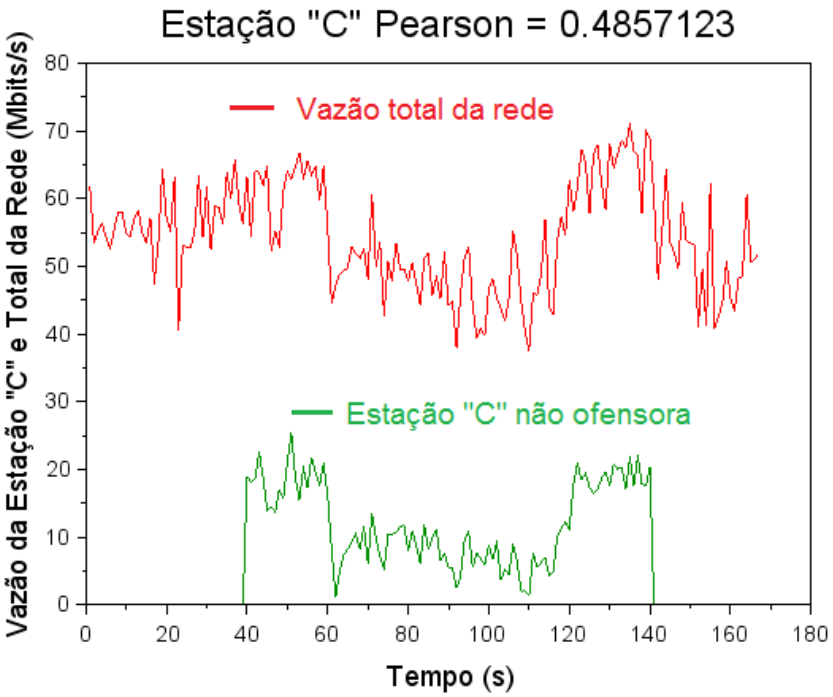


Figura 32: Cenário 4 - Destaque da estação “C”, não ofensora

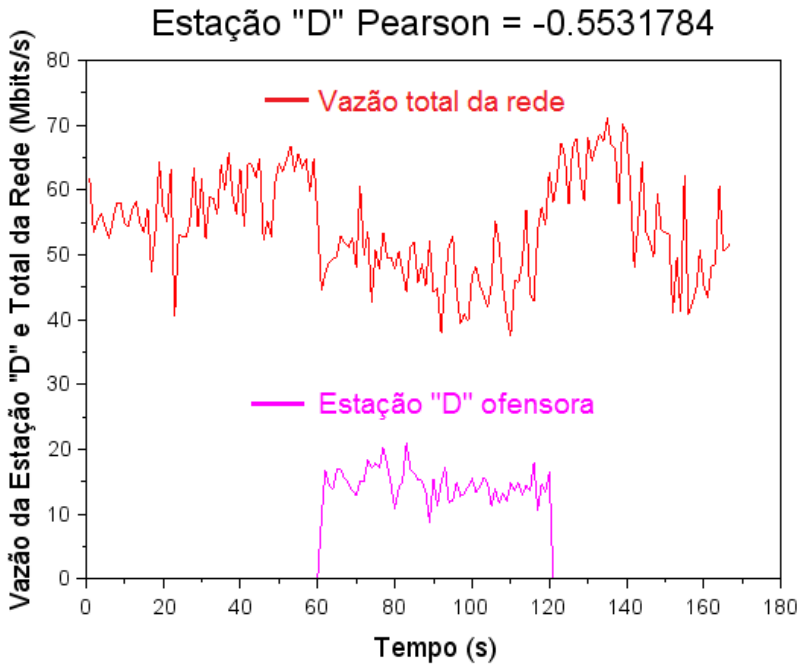


Figura 33: Cenário 4 - Destaque da estação “D”, ofensora

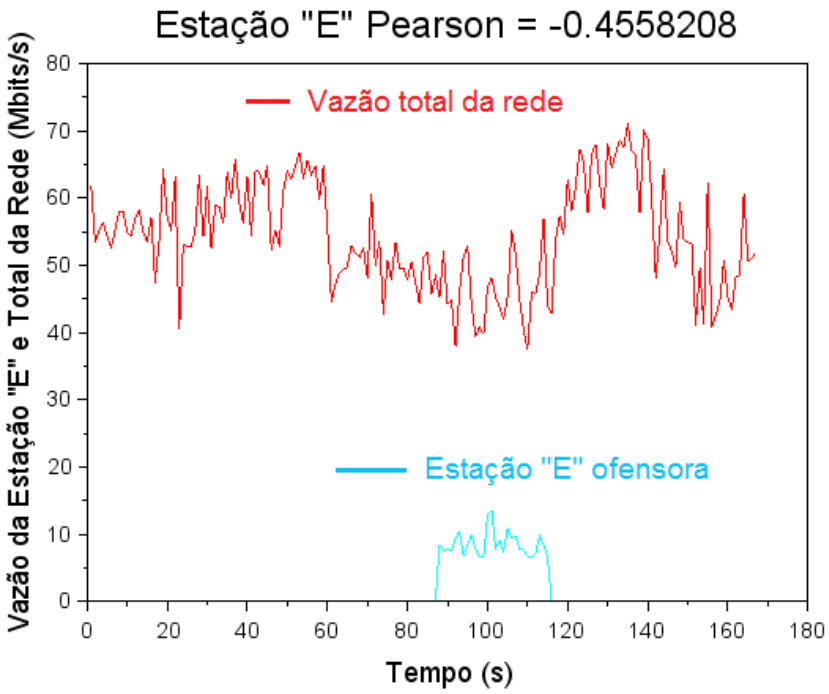


Figura 34: Cenário 4 - Destaque da estação “E”, ofensora

A tabela 4 mostra os CCP para cada estação em cada cenário, comprovando que o método consegue identificar os ofensores. Se o CCP $\rho < 0$, então a estação é ofensora.

Tabela 4: Coeficientes de Correlação de Pearson

Cenário	**Estação**	**Coeficiente** (ρ)
1	A B	0.9153457 -0.4729942
2	A B C	0.9339384 0.9215047 -0.9054236
3	A B C D	0.7714862 0.7571823 0.4406965 -0.7854518
4	A B C D E	0.5214029 0.6291423 0.4857123 -0.5531784 -0.4558208

5.4 Testes de Mitigação da Anomalia

Caso ocorra identificação de estações ofensoras pelo programa “junta.lua”, automaticamente é disparado o script shape.sh que faz *traffic shaping*. A seguir, são apresentados os gráficos das medições realizadas durante este processo.

A cada nova rodada dos scripts integrados (medição, detecção e mitigação), considera-se a inexistência de ofensores, portanto, o *traffic shaping* é removido. Caso sejam identificados ofensores, *o traffic shaping* é renovado.

5.4.1 Resultado da ação do *traffic shaping*

É mostrado o resultado da ação do *traffic shaping* pela figura 37. Inicialmente no intervalo de tempo compreendido entre 0 e 200s, tem-se um cenário com duas

estações conectadas e a anomalia da MAC instalada. Percebe-se visualmente que a estação em azul, tem a sua curva de vazão invertida em relação à curva de vazão total da rede, caracterizando-a como ofensora. O programa de identificação de ofensores junta.lua, calcula o coeficiente de correlação identificando o ofensor (CCP = -0,8581449). Numa segunda etapa, no intervalo de tempo compreendido entre 200 e 350s, foi aplicado o *traffic shaping*. Foi forçada uma queda no nível de vazão da estação ofensora de 8 para 2 Mbits/s. Nota-se uma recuperação imediata na taxa de vazão total da rede. Numa última etapa no intervalo de tempo compreendido entre 350 e 500s, o *traffic shaping* foi retirado somente para demonstrar o retorno da anomalia da MAC. Este é o ciclo de processamento, que pode ser repetido em um laço infinito, ou ser rodado sob demanda, caso haja suspeita de ocorrência da anomalia.

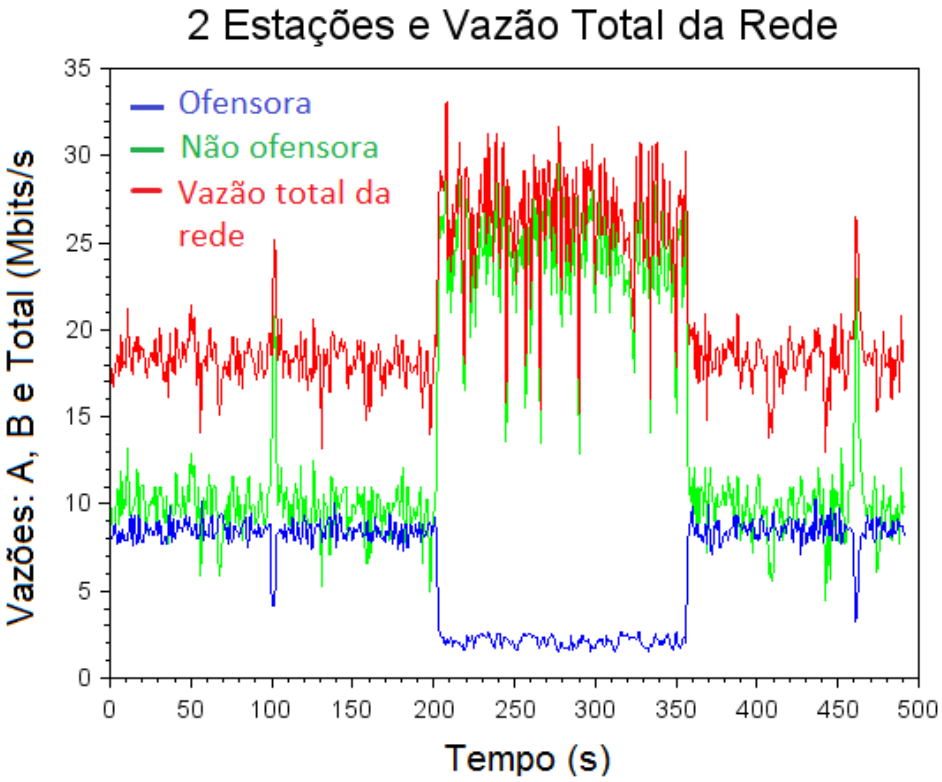


Figura 35: Mitigação de Anomalia com 2 Estações

5.4.2 Cenário de mitigação com duas estações, sendo uma ofensora

É mostrado um cenário de mitigação com duas estações, sendo uma ofensora pela figura 38. No intervalo de tempo compreendido entre 0 e 470s, nenhuma ação é tomada, somente medições de vazão e identificação da estação ofensora. Em uma segunda etapa no intervalo de tempo compreendido entre 470 e 960s, é aplicado o *traffic shaping.* Percebe-se uma recuperação do nível de vazão total da rede. Um novo ciclo se inicia em torno de 960s. O *traffic shaping* é removido e novas medições de vazão são tomadas. Neste último ciclo não foi detectada anomalia e nenhuma ação foi tomada.

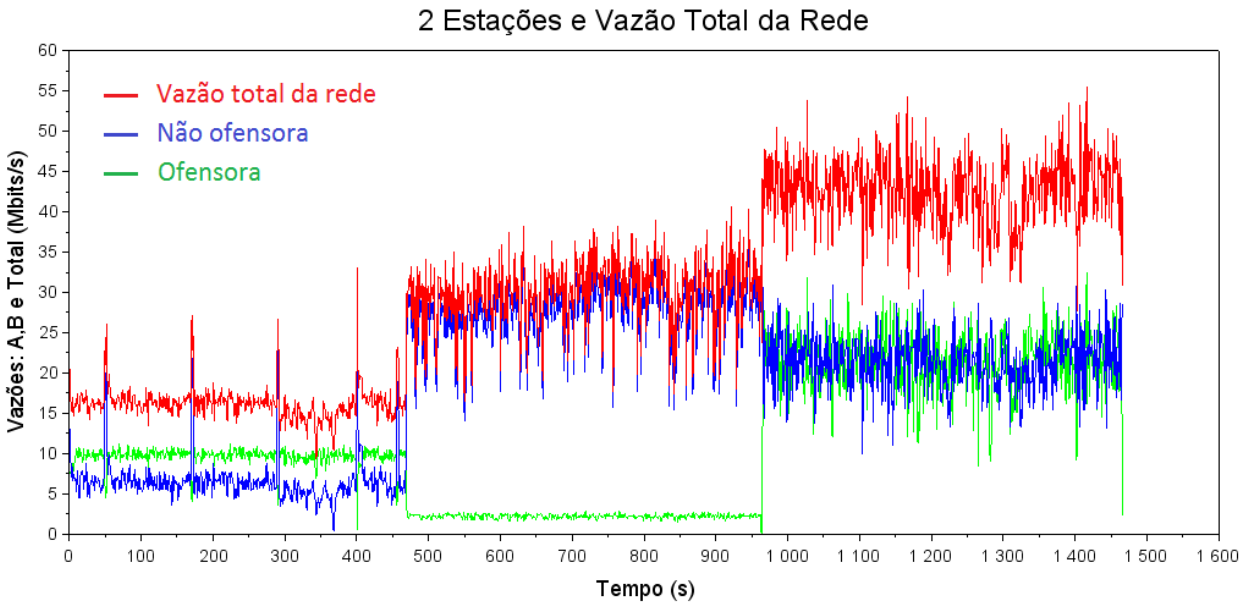


Figura 36: Mitigação com 2 Estações, duas rodadas

5.4.3 Cenário de mitigação com 3 estações, sendo 1 ofensora

É mostrado um cenário de mitigação com 3 estações conectadas, sendo que a estação ofensora aparece em verde pelas figuras 39 e 40. Os scripts integrados estão rodando ciclicamente, sem interrupção. No intervalo de tempo compreendido entre 0 e 350s, nenhuma ação é tomada, somente coleta de dados de vazão. Nota-se um pequeno período de silêncio da estação ofensora entre 180 e 260s quando a vazão total da rede foi recuperada por si mesmo, prontamente. A partir do instante de 350s

(CCP = -0,7532074) é aplicado o *traffic shaping* encerrando-se o primeiro ciclo. O *traffic shaping* permanece ativo durante o intervalo de tempo compreendido entre 350 e 1200s, quando é removido pelo novo ciclo que se inicia. A partir deste instante não foi detectada nova anomalia, não sendo necessária nova mitigação. A figura 40 mostra o destaque de quando teve início o processo todo.

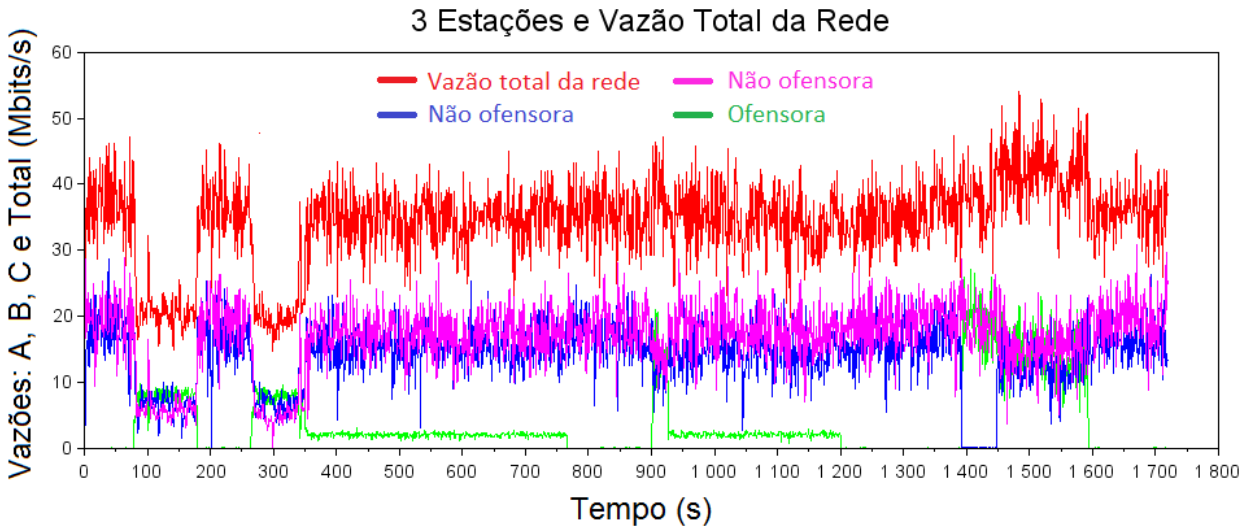


Figura 37: Mitigação com 3 estações, três rodadas

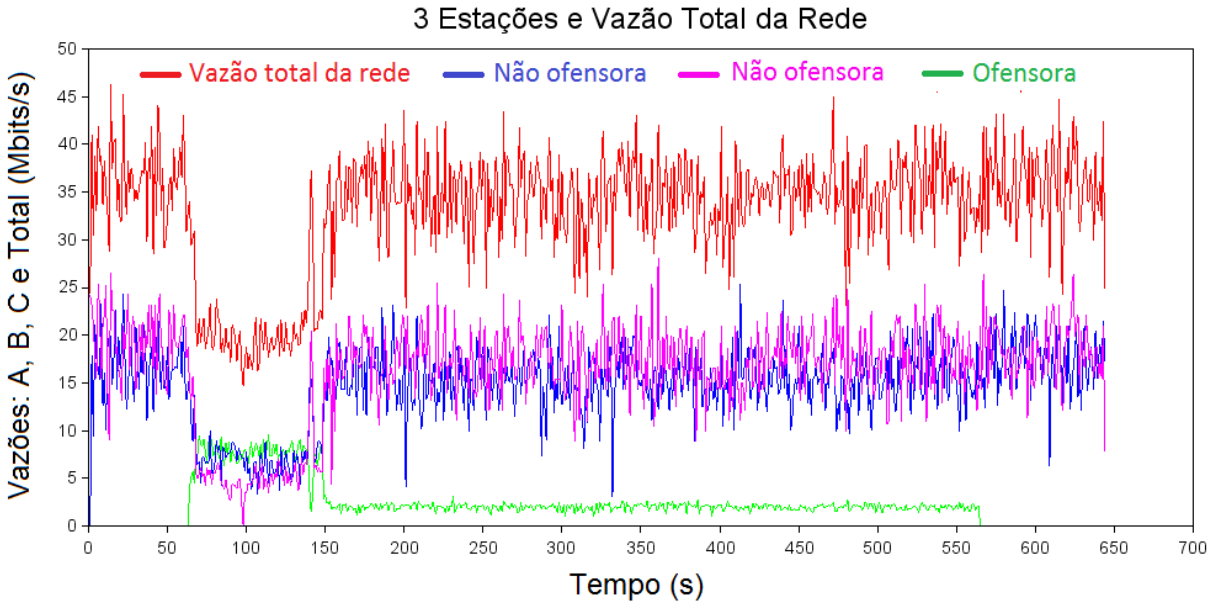


Figura 38: Destaque da Primeira Rodada

5.4.4 Cenário de mitigação com 4 estações, sendo uma ofensora

É mostrado um cenário de mitigação com 4 estações, sendo uma ofensora, pela figura 41. Foi aplicado o *traffic shaping* no instante de 320s, forçando uma queda no nível de vazão da estação ofensora de 5 para 2 Mbits/s. Nota-se uma recuperação imediata do nível de vazão total da rede. No instante de 590s a estação ofensora deixou de transmitir.

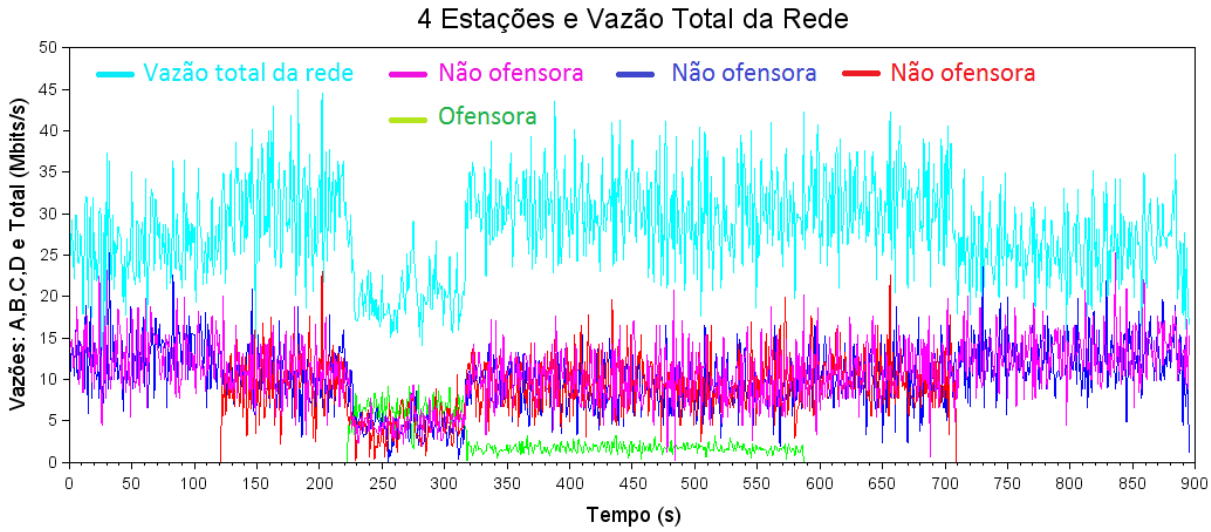


Figura 39: Mitigação com 4 Estações, uma Ofensora

6. CONCLUSÕES

Após decorridos 13 anos desde que a anomalia da MAC foi demonstrada pela primeira vez em (Heusse, 2003), ela ainda está presente no padrão IEEE 802.11. Mesmo com os aperfeiçoamentos constantes visando sempre um aumento na taxa de vazão total da rede, ainda hoje detectamos a anomalia no mais moderno dos equipamentos. Altas taxas de vazão são fundamentais para aplicações de voz sobre IP ou *streaming* de vídeo, tornando-se evidente a necessidade de mitigação de uma eventual ocorrência de anomalia.

Para tanto, foi apresentada uma bancada de tamanho reduzido, baixo custo, com Linux embarcado, portátil e que pode ser utilizada para testes e medições em campo, num ambiente real.

Foram feitas medições de parâmetros de qualidade de operação da rede sem fio tais como: taxa de vazão, intensidade do sinal (RSSI) e relação sinal ruído (SNR). Com base nos dados de vazão, foi desenvolvido um novo método para identificação de ofensores e sua eficácia foi demonstrada com sucesso: método do coeficiente de correlação de Pearson.

A mitigação da anomalia foi então aplicada fazendo-se uso da técnica de *traffic shaping* para se forçar uma redução na taxa de vazão da estação ofensora, minimizando assim os efeitos da anomalia. A estação ofensora, tendo sua taxa de vazão diminuída ainda tem oportunidade de transmitir.

Como resultados finais deste trabalho, apresentou-se um equipamento de baixo custo para acesso à Internet sem fio, que minimiza os efeitos da anomalia da MAC de maneira automática e eficiente. Um novo método para detecção da anomalia e identificação de ofensores foi desenvolvido e testado com sucesso.

Sugestões para desenvolvimentos futuros:

- Testes destes mesmos programas em equipamentos com maior desempenho e versões mais modernas do padrão IEEE 802.11.
- Codificação da lógica dos programas e linguagem C, para que sejam compilados, a fim de melhorar o seu desempenho.
- Testes para a identificação de um maior número de ofensores.
- Testes em redes reais e maiores, com pelo menos 20 estações conectadas.
- Estudar os efeitos do impacto causado pelo *traffic shaping* sobre o tráfego da estação ofensora, antes e depois da mitigação da anomalia. Por exemplo, se a aplicação rodando naquele momento for voz sobre IP (VoIP) ou *streamming* de vídeo, qual a alteração na qualidade das imagens ou da voz?
- Em vez de aplicar shapping fixo de 2Mbits/s, variar o shaping até que seja descoberto um valor máximo, a fim de não impactar a estação ofensora.

7. REFERÊNCIAS

Bianchi, G. Performance Analysis of the IEEE 802.11 Distributed Coordination Function. *IEEE Journal on Selected Areas in Communicatios,* vol.18, no. 3, March 2000. 0733–8716/00$10.00 IEEE.

Branquinho, O. C.; Reggiani, N.; Ferreira, D. M. Mitigating 802.11 MAC Anomaly Using SNR to Control Backoff Contention Window*. International Conference on Wireless and Mobile Communications 2006 ICWMC'06,* pp. 55-55, 2006.

Cheng, Yan-hong; Li Zhi-shu; Xing Jian-chuan; Zhu Li*.* A Novel MAC Mechanism to Resolve 802.11 Performance Anomaly*. Journal of Zhejiang University.* ISSN 1673-565X (Print), ISSN 1862-1775 (Online), 2007.

Comer, Douglas E.; *Internetworking with TCP/IP* volume I. Third Edition. New Jersey: Prentice Hall Inc. 1995. Capítulo 3, página 53. ISBN 0-13-216987-8.

Britto, Dalson F. F.; Silva, J. A. Desvendando os Mistérios do Coeficiente de Correlação de Pearson (r)*. Revista Política Hoje.* Vol 18, n.1, 2009.

DD-WRT Project. *A Linux Based Alternative Open Source Firmware Suitable for a Great Variety of WLAN Routers.* Available at: <http://www.dd-wrt.com>. Accessed on: March, 8th, 2015.

Farber, Betsy; Larson, Ron. *Estatística Aplicada.* 4. ed. São Paulo: Prentice Hall, 2010. Capítulo 9, página 394, tradução de Luciane Ferreira Pauleti Viana.

Garroppo, R. G.; Giordano, S.; Lucetti, S.; Tavanti, L. Providing Air-time Usage Fairness in IEEE 802.11 Networks with the Deficit Transmission Time (DTT) Scheduler. *Springer Science, Wireless Networks,* DOI 10.1007/s11276-006-9201-7 June 15th 2006.

Guirardello, Marcelus. *Política de QoS Com Priorização de Acesso ao Meio para Redes IEEE 802.11.* 2008. 104 páginas. Dissertação (Mestrado em Gestão de Redes de Telecomunicações) – Programa de Pós-Graduação em Engenharia Elétrica, Pontífícia Universidade Católica de Campinas, 2008.

Hallinan, C. *Embedded Linux Primer: A Practical, Real-World Approach*. Crawfordsville, Indiana USA. Publisher: Prentice Hall, 2006 542p. Print ISBN-10: 0-13-167984-8 Print ISBN-13: 978-0-13-167984-9.

Heusse, M.; Rousseau F.; Berger-Sabbatel G.; Duda A. Performance Anomaly of 802.11b. *INFOCOM Twenty-Second Annual Joint Conference of the IEEE.* Grenoble, France, 2003. Computer and Communications. IEEE Societes vol 2 pag. 836-843.

Hubert, B. *Linux Advanced Routing & Traffic Control.* Available at <http://www.lartc.org> Last visited in March, 18th 2015.

IEEE Standards Association. *Overview and Guide to the IEEE 802 LMSC.* September, 2004. PDF. Available at: <http://www.ieee802.org> Accessed on: September 11th 2015.

IEEE Standards Association. *Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.* March, 2012. PDF. Available at: < http://standards.ieee.org/getieee802/download/802.11-2012.pdf> Accessed on: September 11th 2015.

IEEE 802 Comitee Website. Available at: <http://www.ieee802.org> Accessed on: September 11 th 2015.

IEEE 802 Standards <http://standards.ieee.org/about/get/> Accessed on: September 11 th 2015.

Ierusalimschy, R. *Lua The Programming Language 20 years.* Available at: <http://www.lua.org/>. Accessed on: May, 19th, 2015.

ISO, International Standards Organization. *ISO/IEC 7498-4 Internationtal Standard, Information processing systems, Open Systems Interconnection, Basic Reference Model, Part 4 Management framework.* first edition 1989-11-15. Available at: <http://standards.iso.org/ittf/licence.html>. Accessed on: December, 24th, 2015.

Ingle, C.P.; Daryapurkar, R. Performance Analysis of Embedded Linux in Embedded System. *International Journal of Innovative Technology and Exploring Engineering (IJITEE).* ISSN: 2278-3075, Volume-2, Issue-2, January 2013.

Kim, J.; Lee, I. 802.11 WLAN: History and New Enabling MIMO Techniques for Next Generation Standards. *IEEE Communications Magazine.* March 2015. 0163-6804/15.

Kurose, James F., Ross, Keith W.; *Redes de Computadores e a Internet: Uma Abordagem Top-Down.* 3ª edição. São Paulo: Pearson Education do Brasil. 2006. Capítulo 3, página 193. ISBN 85-88639-18-1

Lopez-Aguilera, E.; Casademont, J; Villegas, E. G. A Study on the Influence of Transmission Errors on WLAN IEEE 802.11 MAC Performance. *Wireless Communications and Mobile Computing,* 2010. DOI: 10.1001/wcm.934. Available at <wileyonlinelibrary.com>, Accessed on August, 2015.

P. LORENZ. *QoS in Next Generation Networks.* Revista Telecomunicações, Vol. 1, n. 2, 2009.

Marques, Claurem Paulus Ceolin. *Identificação de Ofensores via Análise da Sensibilidade de Estações na Vazão de Redes IEEE 802.11*. 2013. 96 páginas. Dissertação (Mestrado em Gestão de Redes de Telecomunicações) Programa

de Pós-Graduação em Engenharia Elétrica, Pontífícia Universidade Católica de Campinas. Disponível em: <http://www.bibliotecadigital.puc-campinas.edu.br/tde_busca/>. Acessado em: 8 de março de 2014.

OpenWRT Project. *Wireless Freedom.* Available at <https://openwrt.org.> Accessed on: March, 19th, 2015.

Palazzi, C. E.; Brunati, M.; Roccetti, M. An OpenWRT Solution for Future Wireless Homes. *IEEE International Conference on Multimedia and Expo (ICME),* Suntec City, Singapore, 19-23 July 2010. ISSN: 1945-7871.

Peris, Arturo José Fenile. *Controle de Vazão em Rede IEEE 802.11 com Presença de Ofensores.* 2012. 131 páginas. Dissertação (Mestrado em Gestão de Redes de Telecomunicações) Programa de Pós-Graduação em Engenharia Elétrica, Pontífícia Universidade Católica de Campinas. Disponível em: <http://www.bibliotecadigital.puc-campinas.edu.br/tde_busca/>. Acessado em: 8 de março de 2014.

RAPPAPORT, T. S. *Wirelless Communications Principles and Practice*. Upper Saddle River: Prentice Hall. 2002.

Sandvine Intelligent Broadband Networks. *Global Internet Phenomena Report.* Fall 2011. Available at: <https://dwmw.files.wordpress.com/2011/11/sandvine-global-internet-phenomena-report-fall-2011.pdf> Acessed on: December, 24th 2015.

SciLab *Software*. Disponível em: <http://www.scilab.org/>. Acessada em 10 de novembro de 2015.

Shrivastava, V.; Rayanchu, S.; Yoon, J.; Banerjee, S. 802.11n Under the Microscope. *IMC '08 Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement.* 2008. Pages 105-110. ISBN: 978-1-60558-334-1.

WEB page of Iperf. *Iperf User Documentation.* Available at: <https://iperf.fr/>. Accessed on: March, 19th, 2015.

Zhang, L.; Senac, P.; Lochin, E.; Lacan, J.; Diaz, M. Cross-layer Based Erasure Code to Reduce the 802.11 Performance Anomaly: When FEC meets ARF. *MobiWac '08 - Proceedings of the 6th ACM International Symposium on Mobility Management and Wireless Access.* New York, NY, USA. 2008, Pages 119-124, ISBN: 978-1-60558-055-5

8. APÊNDICES

8.1 Scripts para Medições

taxa_m

```
#!/bin/ash
# Argemiro Bevilacqua – PUCC Mestrado Profissional, 14 de janeiro de 2015.
#
# Coleta e salva os dados de vazão de todas as estações conectadas.
# Estamos medindo apenas o tráfego originário das estações cliente, isto
# é, "rx bytes" segundo o comando "iw". Estas medidas são a quantidade de
# bytes por segundo, transmitidos de cada estação cliente para o ponto de
# acesso.
# O tráfico originado no ponto de acesso com destino às estações conectadas,
# não é objeto de nosso estudo.
# É necessário rodar o script "macaddr" antes de rodar este, a fim de
# atualizar a lista de estações conectadas.
# Um argumento da linha de comando é o número de iterações. Cada iteração
# leva um segundo. Após obter uma lista contendo os MAC addresses das
# estações conectadas, este script dispara uma instância do script "taxa_f"
# para cada estação.
# Este é apenas o script principal que gerencia tudo. Todo o trabalho de
# leitura e gravação dos dados é feito pelas instância do script "taxa_f".
# Após terminadas as medições é chamado outro script: "junta.lua"

HOMEDIR="/tmp/scripts"

if [ $# != 1 ]
then
   echo "USAGE: $0 {number|kill}"
   echo "       One argument is mandatory:"
   echo "       \"kill\": terminates everything that is running."
   echo "       \"number\": starts the program and run it"
   echo "       for the given number of seconds."
   exit 3
fi

ITERATIONS=$1

if [ $ITERATIONS == "kill" ]
then
   for var in `ps | grep taxa_f | grep -v grep | awk '{print $1}'`
   do
      echo "killing $var"
      kill "$var"
   done
   exit 2
fi

NOFSTAS=`$HOMEDIR/nofstas`
if [ $NOFSTAS -eq 0 ]
then
   exit 1
```

```
fi

for var in `$HOMEDIR/macaddr`
do
   $HOMEDIR/taxa_f $var $ITERATIONS &
done

# Aqui tem que dormir o tempo que foi especificado na linha
# de comando, pois os processos filhos (taxa_f) ainda estao
# coletando as medidas. Vamos dormir 10% a mais, só para
# garantir que os arquivos gerados por taxa_f já estão
# salvos e fechados.

SLEEP=`expr $1 + $1 / 10`
sleep $SLEEP

$HOMEDIR/junta.lua $1 *-*-*-*-*-*

# Limpa os arquivos antigos com o formato "*-*-*-*-*-*", caso existam,
# movendo-os para um sub-diretório com nome composto de data e hora
# correntes.

DATE=`date +%y-%m-%d-%H_%M_%S`
ls $HOMEDIR/*-*-*-*-*-* > /dev/null 2>&1



if [ $? -eq 0 ]               # Teste para ver se existe algum arquivo
antigo.
then
   mkdir $HOMEDIR/$DATE
   mv *-*-*-*-*-* $HOMEDIR/$DATE
fi

exit 0
```

taxa_f

```
#!/bin/ash

# Argemiro Bevilacqua, PUCC Mestrado Profissional, 13 de janeiro de 2015.
# Salva a leitura dos dados de vazão de uma estação conectada ao ponto de
# acesso. O MAC address da estação é dado como argumento número 1 na
# linha de comando. O argumento número 2 é o número de iterações.
# Demora um segundo para rodar uma iteração deste script, porque lemos a
# quantidade de bytes transmitidos, esperamos um segundo, lemos novamente
# e fazemos a diferença entre as duas leituras. Esta diferença
# multiplicada por 8 será a vazão em bits por segundo. O resultado é
# então salvo em uma linha adicionada à um arquivo cujo nome é o MAC
# address da estação sendo medida.

BYTES_BEFORE=0
BYTES_AFTER=0
```

```
HOMEDIR="/tmp/scripts"
MAC=$1
FILENAME=`echo $MAC | tr ":" "-"`
ITERATIONS=$2
TIME1=""
TIME2=""

while [ $ITERATIONS -gt 0 ]
do
  BYTES_BEFORE=`iw dev wlan0 station get $MAC|awk '/rx bytes:/ {print $3}'`
  #  Inicio do sleep home made
  TIME1=`date +%y%m%d%H%M%S`
  TIME2=$TIME1
  while [ $TIME2 -eq $TIME1 ]
  do
     TIME2=`date +%y%m%d%H%M%S`
  done
  #  Fim do sleep home made



  BYTES_AFTER=`iw dev wlan0 station get $MAC|awk '/rx bytes:/ {print $3}'`
  if [ $BYTES_BEFORE -gt 0 -a $BYTES_AFTER -gt 0 ]
  then
     echo $TIME2 $BYTES_BEFORE $BYTES_AFTER >> $FILENAME
  fi
  ITERATIONS=`expr $ITERATIONS - 1`
done

exit 0
```

macaddr

```
#!/bin/ash

# Argemiro Bevilacqua, PUCC Mestrado Profissional, 09 de julho de 2014.
# Retorna linhas de texto, cada linha contém um endereço MAC de uma
# estação conectada. É um script auxiliar, principalmente chamado
# por outros como "taxa_m".

HOMEDIR=/tmp/scripts
COUNT=0

for var in `iw dev wlan0 station dump|awk '$1 == "Station" {print $2}'`
do
```

```
   echo $var
   COUNT=`expr $COUNT + 1`
done

if [ $COUNT -eq 0 ]
then
   exit 1
fi

exit 0
```

plot.sce

```
// Script em linguagem do SciLab (compatível com o MathLab). Tem a
// finalidade de ler um arquivo de dados gerados pelo programa
// "junta.lua" e plotar um gráfico dentro do programa SciLab,
// contendo as curvas de vazão de todas as estações medidas mais
// a curva da vazão total da rede.

clear

arquivo=input("Nome do arquivo contendo a matriz de entrada? ","string")

filename=fullfile(arquivo)  // file handle para o diretório corrente.
// Cada linha deste arquivo tem que conter pares ordenados de números
// Reais, separados por vírgula. Podemos ter 2, 3, 4, 5 ou 6 colunas,
// caso sejam 2, 3, 4, ou 5 estações Wi-Fi. Cada coluna contém a vazão
// de uma estação Wi-Fi. A última coluna é a vazão total da rede, ou
// seja a soma de todas as colunas anteriores.

B=csvRead(filename,";");
set("current_figure",99)
tamanho=(min(size(B)));
mprintf("%d\n",tamanho)

if tamanho == 3 then
    j=0;
    for k=1:max(size(B));
        if B(k,1) >= 0 & B(k,2) >= 0 & B(k,3) >=0 then
            j=j+1;
            A(j,1)=B(k,1); A(j,2)=B(k,2); A(j,3)=B(k,3);
        end
    end
    A=A/1000000;
    plot(A(:,1),'green');
    plot(A(:,2),'blue');
    plot(A(:,3),'red');
    xlabel("Tempo (s)", "fontsize", 4, "color", "red");
    ylabel("Vazões: A,B e Total (Mbits/s)", "fontsize",4, "color", "blue");
    a=get("current_axes");
    t=a.title;
    t.foreground=9;
```

```
    t.font_size=5;
    t.font_style=6;
    t.text="2 Estações e Vazão Total da Rede";
end

if tamanho == 4 then
    j=0;
    for k=1:max(size(B));
        if B(k,1) >= 0 & B(k,2) >= 0 & B(k,3) >= 0 & B(k,4) >= 0 then
            j=j+1;
            A(j,1)=B(k,1); A(j,2)=B(k,2); A(j,3)=B(k,3); A(j,4)=B(k,4);
        end
    end
    A=A/1000000;
    plot(A(:,1),'green');
    plot(A(:,2),'blue');
    plot(A(:,3),'black');
    plot(A(:,4),'red');
    xlabel("Tempo (s)", "fontsize", 4, "color", "red");
    ylabel("Vazões: A,B,C e Total (Mbits/s)", "fontsize",4, "color", "blue");
    a=get("current_axes");
    t=a.title;
    t.foreground=9;
    t.font_size=5;
    t.font_style=6;
    t.text="3 Estações e Vazão Total da Rede";
end

if tamanho == 5 then
    j=0;
    for k=1:max(size(B));
        if B(k,1) >= 0 & B(k,2) >= 0 & B(k,3) >= 0 & B(k,4) >= 0 & B(k,5) >=
0 then
            j=j+1;
            A(j,1)=B(k,1); A(j,2)=B(k,2); A(j,3)=B(k,3); A(j,4)=B(k,4);
A(j,5)=B(k,5);
        end
    end
    A=A/1000000;
    plot(A(:,1),'green');
    plot(A(:,2),'blue');
    plot(A(:,3),'black');
    plot(A(:,4),'red');
    plot(A(:,5),'magenta');
    xlabel("Tempo (s)", "fontsize", 4, "color", "red");
    ylabel("Vazões: A,B,C,D e Total (Mbits/s)", "fontsize",4, "color",
"blue");
    a=get("current_axes");
    t=a.title;
    t.foreground=9;
    t.font_size=5;
    t.font_style=6;
    t.text="4 Estações e Vazão Total da Rede";
end

if tamanho == 6 then
```

```
    j=0;
    for k=1:max(size(B));
        if B(k,1) >= 0 & B(k,2) >= 0 & B(k,3) >= 0 & B(k,4) >= 0 & B(k,5) >=
0 & B(k,6) >= 0 then
            j=j+1;
            A(j,1)=B(k,1); A(j,2)=B(k,2); A(j,3)=B(k,3); A(j,4)=B(k,4);
A(j,5)=B(k,5); A(j,6)=B(k,6);
        end
    end
    A=A/1000000;
    plot(A(:,1),'green');
    plot(A(:,2),'blue');
    plot(A(:,3),'black');
    plot(A(:,4),'red');
    plot(A(:,5),'magenta');
    plot(A(:,6),'cyan');
    xlabel("Tempo (s)", "fontsize", 4, "color", "red");
    ylabel("Vazões: A,B,C,D,E e Total (Mbits/s)", "fontsize",4, "color",
"blue");
    a=get("current_axes");
    t=a.title;
    t.foreground=9;
    t.font_size=5;
    t.font_style=6;
    t.text="5 Estações e Vazão Total da Rede";
end
```

corr.sce

```
// Script em linguagem do SciLab. Tem a função de ler um arquivo contendo
// pares ordenados que são: vazão de uma estação e vazão total da rede.
// Cria uma matriz de duas colunas na memória do SciLab. Chama outro
// script em linguagem do SciLab (PEARSON.sci), que calcula o Coeficiente de
// Correlação de Pearson. Após isto, plota um gráfico cartesiano contendo os
// dados de vazão da estação em comparação com os dados de vazão total da
// rede. No cabeçalho é impresso o Coeficiente de Correlação de Pearson.
// São produzidos dois gráficos: Vazão x Tempo e Dispersão de Pearson.

clear
exec 'PEARSON.sci';  // carrega a função na memória.

arquivo=input("Nome do arquivo contendo a matriz de entrada? ","string")
letra=input("Letra da Estação? ","string")

filename=fullfile(arquivo)  // file handle para o diretório corrente.
// Cada linha deste arquivo tem que ser um par ordenado de números
// Reais, separados por vírgula.

B=csvRead(filename,";");

j=0;
for k=1:max(size(B));
```

```
    if B(k,1) >= 0 & B(k,2) >= 0 then
            j=j+1;
            A(j,1)=B(k,1); A(j,2)=B(k,2);
    end
end

mprintf("R=%5.4f\n",PEARSON(A));

set("current_figure",1)
A=A/1000000;
plot(A(:,1),A(:,2),'o');

titulo=strcat(["Vazão da Estação """,letra]);
titulo=strcat([titulo,""" (Mbits/s)"]);
xlabel(titulo, "fontsize", 4, "color", "red");
ylabel("Vazão Total da Rede (Mbits/s)", "fontsize",4, "color", "blue");

a=get("current_axes");
t=a.title;
t.foreground=9;
t.font_size=5;
t.font_style=6;
titulo="Diagrama de Dispersão, Pearson = ";
s=string(PEARSON(A));
result=strcat([titulo,s]);
t.text=result;

set("current_figure",2)
plot(A);
xlabel("Tempo (s)", "fontsize", 4, "color", "red");
titulo="Vazão da Estação """;
titulo=strcat([titulo,letra]);
titulo=strcat([titulo,""" e Total da Rede (Mbits/s)"]);
ylabel(titulo, "fontsize",4, "color", "blue");

a=get("current_axes");
t=a.title;
t.foreground=9;
t.font_size=5;
t.font_style=6;
titulo="Estação """;
titulo=strcat([titulo,letra]);
titulo=strcat([titulo,""" Pearson = "]);
s=string(PEARSON(A));
result=strcat([titulo,s]);
t.text=result;
```

PEARSON.sci

```
// Programa escrito em linguagem do SciLab (compatível com o MathLab)
// Estatística Aplicada, 4ª edição
// Larson, Ron; Farber, Betsy
// Pearson Education do Brasil
```

```
// Exemplo 4, página 399.
//
// Esta é uma função que calcula o Coeficiente de Correlação
// Produto-Momento (r) de Pearson.
// Tem como argumento uma matriz bidimensional contendo pares
// ordenados representando dois vetores a serem testados sobre
// correlação.
// Retorna como saída um número entre -1 e 1. Veja a interpretação
// deste resultado no livro. Veja também a equação.

// A lógica de nossa função, passo-a-passo:
// 1. Encontre a soma dos valores de x, salve em "Sx".
// 2. Encontre a soma dos valores de y, salve em "Sy".
// 3. Multiplique cada valor x por seu valor y correspondente
//    e encontre a soma. salve em "Sxy".
// 4. Faça o quadrado de cada valor x e encontre a soma, salve em "Sxx".
// 5. faça o quadrado de cada valor y e encontre a soma. salve em "Syy".
// 6. Use essas 5 somas para calcular o coeficiente de correlação "R".
//    A seguinte equação será utilizada:
//    R = n*Sxy-Sx*Sy / sqrt(n*Sxx-(Sx^2)) * sqrt(n*Syy-(Sy^2))
// A coluna 1 da matriz A será o nosso "x" e a coluna 2 o nosso "y".

function [R]=PEARSON(A)
    n = max(size(A));
    Sx=0; Sy=0; Sxy=0; Sxx=0; Syy=0; // Somas de x, y, xy, x^2 e y^2.
    Num=0; Den1=0; Den2=0; Den=0; // Numerador e Denominador com 2 partes.
    for i=1:n
        Sx=Sx+A(i,1); Sy=Sy+A(i,2); Sxy=Sxy+A(i,1)*A(i,2);
        Sxx=Sxx+(A(i,1))^2; Syy=Syy+(A(i,2)^2);
    end
    Num=n*Sxy-Sx*Sy;
    Den1=n*Sxx-Sx^2;  // Ainda tem que extrair a raiz quadrada
    Den2=n*Syy-Sy^2;  // Ainda tem que extrari a raiz quadrada
    Den1=sqrt(Den1); Den2=sqrt(Den2); Den=0; Den=Den1*Den2;
    R=Num/Den;
endfunction
```

8.2 Scripts para Cálculo do Coeficiente da Correlação de Pearson

Após as medições serem feitas pelos programas em linguagem Bourne Shell “taxa_m” e “taxa_f”, são produzidos arquivos contendo os dados de vazão medidos, um arquivo para cada estação. O script em linguagem Lua a seguir, junta todos estes arquivos em um só, para que seja possível obter a soma da vazão total da rede. Tendo a vazão total da rede, é possível calcular o Coeficiente de Correlação de Pearson entre cada uma das estações e a vazão total da rede. São produzidos também vários outros arquivos, um para cada estação, contendo a respectiva vazão junto com a vazão total da rede, que servirão para plotar os gráficos no SciLab. Podemos ter arquivos de até 15 estações. Com base no Coeficiente de

Correlação de Pearson calculado, o script a seguir poderá ou não chamar o script que faz *traffic shaping.*

junta.lua

```
#!/usr/bin/lua

-- Este programa foi feito para rodar no Linux embarcado OpenWRT em
-- linguagem Lua. Faz processamento com vetores e matrizes. Seu objetivo
-- eh identificar um ofensor dentro de uma rede Wi-Fi, utilizando-se
-- do Coeficiente de Correlacao de Pearson  (CCP). Aplicamos o calculo do
-- CCP entre a vazao de uma estacao e a vazao total da rede. Este programa
-- tem capacidade para calcular ateh 15 estacoes.
-- O programa "junta.lua" faz um "merge", isto eh, junta todos os arquivos
-- de dados, cujo formato eh:
-- Data         antes depois
-- 141001160454 96030 96142
-- 141001160455 96142 96336
-- 141001160456 96336 96553
-- 141001160457 96679 97695
-- Data: Quando foi feita a leitura (Ano, Mes, Dia, Hora, Minuto e Segundo).
-- antes:  quantidade de bits transmitidos antes  de 1 segundo.
-- depois: quantidade de bits transmitidos depois de 1 segundo.
-- Se fizermos a subtracao de depois-antes, obteremos a quantidade de bits
-- transmitidos em 1 segundo, ou seja, a taxa de bits/s.
-- Como rodar este programa em linha de comando: lua junta.lua 10 *-*-*-*-*-*
-- O primeiro argumento (10), eh o numero maximo de linhas que os arquivos
-- contem. O segundo argumento (*-*-*-*-*-*) eh uma lista com os nomes de
-- todos os arquivos de entrada. Pode ser usado o caracter curinga "*", que
-- sera expandido para uma lista de nomes de arquivos. Os nomes de arquivos
-- de entrada tem o formato do MAC address, por exemplo: e0-f1-ee-ab-01-dd.
-- O resultado do "merge" eh uma matriz de 17 colunas por "n" linhas,
-- onde "n" eh o numero de linhas dos arquivos de entrada. No estagio
-- de leitura de dados, eh feita uma leitura  por segundo, portanto o
-- numero de linhas pode ser entendido como o numero de segundos.
-- O formato da matriz eh:
--    de 01 a 15: Vazao das estacoes de 1 a 15
--    16: Data (veja formato acima)
--    17: Vazao total da rede (soma das vazoes das estacoes de 1 a 15).
-- O numero de colunas de nossa matriz reservado para as estacoes eh
-- 15, portanto aceitaremos no maximo 15 arquivos de entrada. Conte mais
-- 1 argumento que eh o numero de linhas dos arquivos.

if #arg > 16 then  -- Nao aceita mais do que 16 argumentos, ou 15 arquivos.
   return 1
end
```

```
-- Criacao e inicializacao das matrizes T e U (U eh copia de T)
-- Cada linha de nossa matriz tera este formato: 17 campos numericos.

local T =
{
{999999999, 999999999, 999999999, 999999999, 999999999, 999999999, 999999999,
999999999, 999999999, 999999999, 999999999, 999999999, 999999999, 999999999,
999999999, 0, 0}
}

-- Copia (limpa e filtrada) da matriz T.
local U = { {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0} }

-- A matriz T tem o mesmo numero de linhas dos arquivos de entrada
-- sendo lidos.
-- arg[1] Quantidade de linhas contidas no maior dos arquivos.
for nlinhas=1,arg[1]  do
   T[nlinhas]={999999999,999999999,999999999,999999999,999999999,999999999,
               999999999,999999999,999999999,999999999,999999999,999999999,
               999999999,999999999,999999999,0,0}
end -- Reservamos espaco de memoria para todas as linhas.

-- Lendo os arquivos de entrada e carregando-os na matriz T.
-- Haverao falhas em T, pois faltarao algumas linhas de leitura de vazão
-- nos arquivos de entrada, isto eh, saltos nos segundos da coluna de data.
-- De arg[2] ateh arg[16] temos nomes de arquivos a serem lidos.
for arquivo=2,#arg  do
   local countlin=0
   -- leh uma linha e armazena em "cadeia".
   for cadeia in io.lines(arg[arquivo]) do
      countlin=countlin+1
      -- "a" indexa o inicio da primeira string.
      local a = string.find(cadeia,"%s")
      --"b" indexa o inicio da segunda string.
      local b = string.find(cadeia,"%s",a+1)
      --"c" indexa o inicio da terceira string.
      local c = string.find(cadeia,"%s",b+1)
      data = string.sub(cadeia,1,a)
      antes = string.sub(cadeia,a,b)
      depois = string.sub(cadeia,b,c)
      taxa = (depois - antes) * 8
      if T[countlin][16] == 0 then -- data == 0. Linha vazia, pode inserir!
         T[countlin][arquivo-1]=taxa -- posicoes para a taxa: de 1 a 15.
         T[countlin][16]=data -- Posição 16 data e 17 vazao total da rede.
      elseif T[countlin][16] == data then -- Data bateu, pode inserir!
         T[countlin][arquivo-1]=taxa
      Else
         -- Procura uma linha na matriz, cuja data eh igual a "data" lida.
```

```
         for var=1,#T
         do
            if T[var][16]==data then
               T[var][arquivo-1]=taxa
            end
         end
      end
   end
end


-- Limpeza de linhas invalidas na matriz T:
-- Algumas estacoes nao tem medicoes em todos os segundos.
-- Se examinarmos os arquivos contendo as medicoes, veremos saltos na
-- coluna "Data" (veja acima a descricaoo do formato do arquivo).
-- Sendo assim, estes vazios estarao preenchidos por "99999999" que sao
-- os valores iniciais que nao foram atualizados. Agora vamos fazer uma
-- limpeza nestas linhas da matriz "T", pois se pelo menos uma estacaoo
-- nao possuir medicaoo naquele segundo, toda a linha serah eliminada,
-- para que a vazao total da rede nao apresente valores irreais (estacao
-- sem medida somaria "999999999" ou incognita). Depois de limpa a matriz
-- T, serah gerada como resultado a matriz limpa U e abandonaremos T.
-- Aqui tambem eh feita a soma da vazao total da rede e armazenado
-- o resultado em T[var][17].

j = 0 -- Indexador de linhas para a matriz U.
R = 0 -- Teste para saber se existe somente uma estacao com medicao (alone).
      -- Caso positivo, a linha é eliminada.

-- Vetores usados no calculo do Coeficiente de Correlacaoo de Pearson (CCP):
x = {}  -- x: Tamanho=U#, vazao da estacao com ofensividade sendo analisada
y = {}  -- y: Tamanho=U#, contem a vazao total da rede.

for var=1,#T
do
   R = 0
   for arquivo=2,#arg  -- Soma vazao total da rede.
   do
      if T[var][arquivo-1] > 25000 then
         R = R + 1
      end
      T[var][17] = T[var][17] + T[var][arquivo-1]
   end
   -- Copia de T para U somente linhas que nao tenham
   -- "999999999" (soma < 9000000000)
   if T[var][17] < 900000000 then
      if R > 1 then
         j = j + 1
         -- Insere uma linha na matriz U
```

```
         U[j] = {0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0}
         for i=1,#arg-1
         do
            U[j][i]=T[var][i] -- U recebe a linha de T
         end
         y[j]=T[var][17]
      end
   end
end


-- Carregamento do vetor "x": teremos um vetor "x" para cada uma das
-- estacoes, contendo a sua vazao lida minuto-a-minuto.
-- Chama a funcao que calcula o Coeficiente de Correlacao de Pearson.
-- Pode ocorrer de haver somente uma estacao conectada, ou nenhuma estacao
-- apresentou vazao superior a 25000 bits/s. Entao U sera uma matriz vazia.
-- Se #U == 1, entao todas as linhas lidas foram filtradas e descartadas.

-- Remove todos os "Traffic Shapping" que porventura estejam rodando.
-- Caso sejam detectados novos ofensores ou a permanencia dos antigos,
-- novos "Traffic Shapping" serao criados, conforme abaixo, um para cada
-- estacao ofensora.
os.execute("date >> /tmp/scripts/log")
local comando="/tmp/scripts/shape.sh".." "."stop".">>/tmp/scripts/log 2>&1"
local logstring="echo "..comando.." "..">>".."/tmp/scripts/log"
os.execute(logstring)
os.execute(comando)

-- apenas carrega a funcao "pearson.lua" na memoria
dofile("/tmp/scripts/pearson.lua")
-- Calculando Pearson e refazendo todos os "Traffic Shapping".
if #U > 1 then
   local nomearq1="00-00-00-00-00-00" -- contem os CCP das estacoes
   handle1=io.open(nomearq1,"w")
   for estacao=1,#arg-1 -- Nomes de arquivos a serem lidos e carregados.
   do
      local nomearq2=arg[estacao+1]..".csv" --Vazao da rede e da estacao.
      handle2=io.open(nomearq2,"w")
      for nlinhas=1,#U -- carregamento do vetor x.
      do
         x[nlinhas]=U[nlinhas][estacao]
         handle2:write(x[nlinhas]..";"..y[nlinhas].."\n")
      end
      handle2:close()
      local pearson=pearson(x,y)
      handle1:write(arg[estacao+1].." "..pearson.."\n")
      local argumento=arg[estacao+1]; mac=string.gsub(argumento,"-",":")
      if pearson < -0.1 then -- Roda Traffic Shapping para todas ofensoras
         os.execute("date >> /tmp/scripts/log")
```

```
         local comando=
         "/tmp/scripts/shape.sh".." "..mac..">> /tmp/scripts/log 2>&1"
         local logstring="echo "..comando.." "..">>".."/tmp/scripts/log"
         os.execute(logstring)
         os.execute(comando)
      end
   end
   handle1:close()

   -- Cria o arquivo com a vazao de todas as estacoes, mais a vazao total
   -- da rede, somente para fins de construcao de um grafico e documentacao.
   local nomearq3="99-99-99-99-99-99.csv"
   local linha=""
   handle3=io.open(nomearq3,"w")
   for var=1,#U
   do
      for estacao=1,#arg-1
      do
         linha=linha..U[var][estacao]..";"
      end
      linha=linha..y[var].."\n"
      handle3:write(linha); linha=""
   end
   handle3:close()
end
```

corr.lua

```
#!/usr/bin/lua

-- Le um arquivo de entrada contendo pares ordenados e calcula o
-- Coeficiente de Correlacao de Pearson entre os dois vetores.
-- O formato das linhas do arquivo é:
-- 7642968;7642968
-- 16436160;16436160
-- 25356912;25356912
-- ...
-- O nome do arquivo tem que ser passado como argumento na linha
-- de comando que chama o programa.
-- Os números devem ser separados por ";".
-- Vamos por o primeiro vetor no eixo x e o segundo vetor no eixo y
-- Para iniciar os cálculos, vamos primeiramente construir duas tabelas
-- em seguida passamos essas tabelas como argumento a uma funcao que nos
-- retornara o valor do Coeficiente de Correlacao de Pearson.
-- Feito isto, o programa termina.
--
```

```
if #arg ~= 1 then
   print("   \n\tFaltou o nome do arquivo de entrada na linha de comando,")
   print("\tprograma abortado. Forma de uso:");
   print("", arg[0], "<nome_arq.>")
   print("\n\tO formato do arquivo de entrada tem que ser parecido com:")
   print("\t...")
   print("\t16436160;16436160")
   print("\t25356912;25356912")
   print("\t...")
   return 1
end

x = {}
y = {}

for var in io.lines(arg[1])
do
   local a = string.find(var,";")
   table.insert(x,string.sub(var,1,a-1))
   table.insert(y,string.sub(var,a+1,#var))
end

-- apenas carrega a função "pearson.lua" na memoria.
dofile("pearson.lua")
-- chama a função que calcula o Coeficiente de Correlação de Pearson.
print(pearson(x,y))
return 0
```

pearson.lua

```
-- Argemiro Bevilacqua, 27 de Maio de 2015
-- Programa de Mestrado em Gerência de Redes e Telecomuncacoes
-- PUC Campinas
-- Funcao "pearson.lua"
-- Aceita dois vetores como argumento.
-- Os vetores sao pares ordenados.
-- Calcula o Coeficiente de Correlacao de Pearson.

function pearson(x,y)
   local R = 0
   local n = #x  -- #x = #y. Tanto faz: tamanho de x ou de y
   -- Somas de x, y, xy , x^2 e y^2.
   local Sx=0; local Sy=0 local Sxy=0 local Sxx=0 local Syy=0
   -- Numerador e Denominador com 2 partes.
   local Num=0 local Den=0 local Den1=0 local Den2=0

   for i=1,n do
      Sx=Sx+x[i]; Sy=Sy+y[i]; Sxy=Sxy+(x[i]*y[i]);
```

```
      Sxx=Sxx+(x[i])^2; Syy=Syy+(y[i]^2);
   end

   Num=(n*Sxy)-(Sx*Sy)
   Den1=(n*Sxx)-(Sx^2);  -- Ainda tem que extrair a raiz quadrada
   Den2=(n*Syy)-(Sy^2);  -- Ainda tem que extrair a raiz quadrada
   Den1=math.sqrt(Den1); Den2=math.sqrt(Den2); Den=Den1*Den2;
   R=Num/Den;

   return R
end
```

8.3 Script para Execução de Traffic Shaping no Roteador

shape.sh

```
#!/bin/ash

# Pontificia Universidade Catolica de Campinas
# Programa de Mestrado Profissional em Gestão de Redes e Telecomunicações
# Mestrando R.A. 14381834 Argemiro Bevilacqua - 12 de novembro de 2014.
#
# Este programa efetua "traffic shaping" da seguinte maneira:
# Dada uma interface de rede sem fio e um MAC address, aplica uma política
# sobre o trafego entrante, limitando a taxa máxima de transferência a
# um certo limite. Comecaremos a descartar pacotes se recebermos trafego
# acima do valor especificado. Usado o comando "iptables" do Linux para
# selecionar e marcar os pacotes entrantes com o número "1", caso o MAC
# address for igual àquele especificado na linha de comando.
# Criado um filtro que reconhece os pacotes marcados com o numero "1"
# por intermedio do subcomando "handle 1 fw" que aplica a politica
# descrita acima.
#
# Por favor, leia http://www.lartc.org para ter uma completa descricao dos
# comandos "iptables" e "tc" usados aqui.

usage(){
   echo "$0: USAGE: $0 {00:00:00:00:00:00|show|statistics|stop} <ENTER>"
}

# Inicializacao das variaveis de ambiente e os varios caminhos de comandos.
TC=/usr/sbin/tc
IW=/usr/sbin/iw
MAC=0
SIZE=0
```

```
RETURN_CODE=0
DEV=wlan0
SRC=0              # Endereco IP do suposto ofensor.
IP=/usr/sbin/ip

# Fazendo validacao da linha de comando para ver se foi digitado
# um MAC address valido, ou as opcoes "show" e "clean".
if [ $# -ne 1 ]
then
   echo $0: "Numero insuficiente de argumentos."
   usage
   exit 1
fi

if [ "$1" = "show" ]
then
   # lista todas as filas "qdiscs" da interface.
   $TC qdisc ls dev wlan0
   # listas as filas do tipo "ingress".
   $TC filter ls dev wlan0 parent ffff:
   exit 0
fi

if [ "$1" = "statistics" ]
then
   $TC -s qdisc ls dev wlan0     # lista estatisticas da interface.
   exit 0
fi

if [ "$1" = "stop" ]
then
   $TC qdisc del dev wlan0 ingres  # Deleta a fila do tipo "ingress".
   exit 0
fi

SIZE=`echo $1 | wc -c`
if [ $SIZE -lt 17 ]
then
   echo $0: "O endereco MAC deve ter tamanho de 17 caracteres.";
   usage;
   exit 2
fi

RETURN_CODE=$?
if [ $RETURN_CODE -ne 0 ]
then
   echo $0: "O argumento tem de conter caracteres \":\"".
   usage;
   exit 3
```

```
fi
MAC=$1

# Verifica de o endereco MAC fornecido pertence a uma estacao conectada.
$IW dev $DEV station dump | awk '/Station/ {print $2}' | grep -q $MAC
RETURN_CODE=$?
if [ $RETURN_CODE -ne 0 ]
then
   echo $0: "A estacao $MAC nao esta conectada.";
   exit 4
fi

# Fim da validacao do endereco MAC. Agora nos temos um endereco MAC
# valido e conectado de uma estacao cliente de Wi-Fi.

# Descobre qual endereco IP esta associado com o endereco MAC dado
SRC=`ip neig | grep $MAC | awk ' $1 !~/:/ {print $1}'`

# Cria uma fila de pacotes entrantes do tipo "qdisc" - queue discipline.
$TC qdisc add dev $DEV handle ffff: ingress

# Cria um filtro que usa o endereco IP para separar os pacotes.
# Aplica uma politica nos pacotes selecionados: Descarta os pacotes
# quando a vazao ultrapassar 2Mbits/s, utilizando um buffer de
# 200 Megabytes.
$TC filter add dev $DEV parent ffff: protocol ip prio 50 u32 match ip \
src $SRC/32 police rate 2mbit buffer 200m drop
```

8.4 Publicações

1. **Embedded Software Platform Used for Detection and Mitigation of a MAC Anomaly in IEEE 802.11n Networks**. AJAS - American Journal of Applied Sciences. Artigo publicado em revista científica internacional em novembro 2015**.** Available at: http://thescipub.com/abstract/10.3844/ajassp.2015.902.916

2. **Especificação de Bancada de Testes em Redes Wi-Fi IEEE 802.11 para Detecção de Anomalia na Camada MAC.** IX Workshop de Pós-Gradução e Pesquisa do Centro Paula Souza, São Paulo, 15 e 16 de Outubro de 2014. Artigo aceito e publicado. Feita a apresentação no Workshop. Disponível em: http://www.centropaulasouza.sp.gov.br/pos-graduacao/workshop-de-pos-graduacao-e-pesquisa/009-workshop-2014/workshop/desenvolvimento-de-tecnologias-e-sistemas.htm

3. **Degradação do Parâmetro de Qualidade *Jitter*: Análise do Meio.** X Workshop de Pós-Gradução e Pesquisa do Centro Paula Souza, São Paulo, 6 a 8 de Outubro de 2014. Artigo aceito e publicado. Feita a apresentação no Workshop. Disponível em: http://www.centropaulasouza.sp.gov.br/pos-graduacao/workshop-de-pos-graduacao-e-pesquisa/010-workshop-2015/workshop/trabalhos/Sistemas_Produtivos/Desenv_Tecn_Sist_Prod/Degrad_Parametro_Qualid_Jitter.pdf

4. **Verificação da Qualidade do Sinal de Redes Wi-Fi.** Simpósio do Programa de Pós-Graduação em Sistemas de Infraestrutura Urbana (SPInfra) – 2014. Resumo aceito e apresentado no Simpósio.

5. **Identificação da Anomalia da MAC em Redes IEEE 802.11.** Resumo aceito e apresentado na *67ª Reunião Anual da SBPC.* Disponível em: http://www.sbpcnet.org.br/livro/67ra/resumos/resumos/5032_19aab4ca24e07ce6888d1e5678aa1a2ac.pdf

6. **Communications Delay and Energy Consumption in IEEE 802.11 Network Stations.** IJCSNS International Journal of Computer Science and Network Security, VOL.15 No.8, August 2015. Artigo publicado em revista científica internacional. Available at: http://paper.ijcsns.org/07_book/201508/20150804.pdf